ANL/NE-15/44

Multi-Physics Demonstration Problem with the SHARP Reactor Simulation Toolkit

Nuclear Engineering Division

About Argonne National Laboratory
Argonne is a U.S. Department of Energy laboratory managed by UChicago Argonne, LLC
under contract DE-AC02-06CH11357. The Laboratory's main facility is outside Chicago,
at 9700 South Cass Avenue, Argonne, Illinois 60439. For information about Argonne
and its pioneering science and technology programs, see www.anl.gov.

DOCUMENT AVAILABILITY

Online Access: U.S. Department of Energy (DOE) reports produced after 1991 and a growing number of pre-1991 documents are available free via DOE's SciTech Connect (http://www.osti.gov/scitech/)

Reports not in digital format may be purchased by the public from the
National Technical Information Service (NTIS):
U.S. Department of Commerce
National Technical Information Service
5301 Shawnee Rd
AlexandrIa, VA 22312
www.ntis.gov
Phone: (800) 553-NTIS (6847) or (703) 605-6000
Fax: (703) 605-6900
Email: **orders@ntis.gov**

Reports not in digital format are available to DOE and DOE contractors from the
Office of Scientific and Technical Information (OSTI):
U.S. Department of Energy
Office of Scientific and Technical Information
P.O. Box 62
Oak Ridge, TN 37831-0062
www.osti.gov
Phone: (865) 576-8401
Fax: (865) 576-5728
Email: **reports@osti.gov**

Disclaimer

This report was prepared as an account of work sponsored by an agency of the United States Government. Neither the United States Government nor any agency thereof, nor UChicago Argonne, LLC, nor any of their employees or officers, makes any warranty, express or implied, or assumes any legal liability or responsibility for the accuracy, completeness, or usefulness of any information, apparatus, product, or process disclosed, or represents that its use would not infringe privately owned rights. Reference herein to any specific commercial product, process, or service by trade name, trademark, manufacturer, or otherwise, does not necessarily constitute or imply its endorsement, recommendation, or favoring by the United States Government or any agency thereof. The views and opinions of document authors expressed herein do not necessarily state or reflect those of the United States Government or any agency thereof, Argonne National Laboratory, or UChicago Argonne, LLC.

ANL-NE-15/44

Multi-Physics Demonstration Problem with the SHARP Reactor Simulation Toolkit

prepared by
E. Merzari, E. Shemon, Y. Yu, J. Thomas
Nuclear Engineering Division, Argonne National Laboratory

A. Obabko, R. Jain, V. Mahadevan
Math and Computer Science Division, Argonne National Laboratory

J. Solberg, R. Ferencz, R. Whitesides
Methods Development Group, Lawrence Livermore National Laboratory

December 21, 2015

ABSTRACT

To design an inherently safe sodium-cooled fast reactor (SFR), it must be demonstrated that he net reactivity coefficient is negative, such any event that causes the core power to increase initially will be quickly followed by a response that tends to decrease the core power and return the reactor to a safe operating condition. This response in the core reactivity is caused by several mechanisms (which may compete with each other), including coolant density changes, the fuel Doppler effect, and changes in core geometry. Simulating the latter mechanism, changes in core geometry, is the focus of the multi-physics demonstration in this report. In particular, the focus is on the focus of radial core expansion caused by the motion of fuel assemblies in response to thermal expansion.

The core restraint system must be carefully designed to ensure that temperature increases induce thermal expansion that results in the fuel assemblies moving outward and thus away from each other. In particular, any fuel assembly distortion and displacement must occur in such a way that the fuel elements—typically located in the central portion of a significantly longer fuel assembly—move away from each other. This induces a negative reactivity response and helps return the reactor to a safe operating condition. In fact, in reactor designs under consideration by the Advanced Reactor Concepts program, core radial expansion is the dominant negative reactivity feedback mechanism.

This expansion phenomenon, which includes the physics of neutronics, thermal hydraulics, and structural mechanics, is challenging to model. In fact, conventional SFR safety analyses do not include the effect of fuel assembly bowing, and the physics are quite loosely coupled in a multi-step simulation procedure. The objective here is to demonstrate a multi-physics modeling and simulation capability that can explicitly predict the deformed core geometry, neutronics feedback with consistent power distributions, and temperature and flow distributions.

SHARP, developed under the NEAMS Reactor Product Line, is an advanced modeling and simulation toolkit for the analysis of nuclear reactors. SHARP is comprised of several components, including physical modeling tools, tools to integrate the physics codes for multi-physics analyses, and a framework to couple the tools together. Physics modules currently include neutronics, thermal hydraulics, and structural mechanics. SHARP empowers designers to analyze transformative reactor concepts with simulation tools that are not limited to available experimental data sets from currently existing reactor designs. By developing the tools to be highly efficient on parallel computing platforms, engineering-scale simulations become practical on high-performance computers currently available within the DOE complex. Development efforts strive to work in tandem with efforts in experimentation, so that the tools are validated to produce accurate results for modeling physical phenomena that have been identified as important for nuclear reactor analysis. By taking this approach, SHARP supports nuclear reactor analysis and design activities for DOE programs and industrial partnerships with trustworthy modeling and simulation tools.

This report describes to employ SHARP to perform a first-of-a-kind analysis of the core radial expansion phenomenon in an SFR. This effort required significant advances in the framework used to drive the coupled simulations, manipulate the mesh in response to the deformation of the geometry, and generate the necessary modified mesh files. Furthermore, the model geometry is fairly complex, and consistent mesh generation for the three physics modules required significant effort. Fully-integrated simulations of a 7-assembly mini-core test problem have been performed, and the results are presented here. Physics models of a full-core model of the Advanced Burner Test Reactor have also been developed for each of the three physics modules. Standalone results of each of the three physics modules for the ABTR are presented here, which provides a demonstration of the feasibility of the fully-integrated simulation.

Table of Contents

LIST OF FIGURES

LIST OF TABLES

1 Introduction

SHARP [1], developed under the NEAMS program, is an advanced modeling and simulation toolkit for the analysis of nuclear reactors. SHARP is comprised of several components, including physical modeling tools, tools to integrate the physics codes for multi-physics analyses, and a set of tools to couple the codes within the MOAB [2] framework. Physics modules currently include the PROTEUS [3] neutronics code, the Nek5000 [4] thermal-hydraulics code, and the Diablo [5] structural mechanics code. The development philosophy for the physics modules is to incorporate as much fundamental physics as possible, rather than developing tools for specific reactor analysis applications. This empowers designers to analyze transformative reactor concepts with simulation tools that are not limited to available experimental data sets from currently existing reactor designs. By developing the tools to be highly efficient on parallel computing platforms, employing millions of processor cores, engineering-scale simulations become practical on high-performance computers currently available at the DOE complex. Development efforts strive to work in tandem with efforts in experimentation, so that the tools are validated to produce accurate results for modeling physical phenomena that have been identified as important for nuclear reactor analysis. By taking this approach, SHARP supports nuclear reactor analysis and design activities for DOE programs and industrial partnerships with trustworthy modeling and simulation tools.

The SHARP multi-physics modeling capability is being demonstrated for the problem of radial core expansion and bowing in a sodium-cooled fast reactor. To design an inherently safe fast reactor, reactivity dependence on radial core expansion must be engineered into the reactor plant to assure a loss of reactivity during transient events. In the advanced SFR concepts currently under consideration by the Advanced Reactor Concepts program, the core is designed to bow outward in response to thermal expansion of the structures in any transient where the core is heating. The grid plate and load pads, which support the core from below and restrain it from the top, respectively, also expand outward. Moreover, the core restraint system is designed such that the fuel assemblies bow outward in the middle, further separating the fuel pins.

When controlled correctly, core expansion causes the fuel assemblies to move farther apart from each other, which has a negative reactivity effect and helps to shut down the reactor. Simulation of this expansion, which is essential to the safety of these reactor concepts, necessitates the coupling of structural mechanics, thermal hydraulics, and structural mechanics.

For safety analyses supported by the Advanced Reactor Concepts program, the radial core expansion phenomena would typically be performed by a multi-step procedure. First, a neutronics perturbation analysis would be performed. The approach here is to analyze the core under two conditions—a reference condition and an expanded condition. The complex geometry of the expanded core cannot be modeled by currently available neutronics codes; the postulated expansion must be uniform everywhere in the core. In fact, the expansion must be uniform both radially and axially, i.e., all assemblies expand by the same amount, and the deformation does not vary along the length of the assembly. The perturbation analysis on this simplified geometry produces a reactivity coefficient, which is an indication of the change in core reactivity per unit of core expansion. This coefficient is then employed in a separate transient analysis with the system code SAS4A/SASSYS-1. The system code predicts the core temperature during the postulated transient scenario, and uses that temperature in a relatively simple model that predicts the motion of the core as it expands. In fact, this system code model is not capable of modeling assembly bowing, but only includes the effect of grid plate and load pad expansion. In this way, the multi-physics phenomena is handled in a loosely coupled way: a series of steady-state neutronics perturbation analyses are performed for a uniform core expansion to generate a reactivity coefficient, followed by a separate system code transient simulation that estimates the transient core deformation and applies this coefficient to predict the reactivity response.

The SHARP analysis that will be described in this report is a first-of-a-kind effort to perform a single integrated simulation of the core undergoing radial thermal expansion while retaining the full geometric detail necessary to model physical phenomena at the continuum scale. This simulation employs the neutronics, thermal-hydraulics, and structural mechanics simulations simultaneously, with no need for offline perturbation analyses or passing coefficients between the physics codes. The reactivity feedback from core deformation is being predicted explicitly. The power distribution from nuclear fission is predicted by the neutronics module, which influences the temperature field in the thermal-hydraulics module. Structure temperature profiles predicted by the thermal-hydraulics module drive deformations in the structural mechanics module, and the neutronics and thermal-hydraulics simulations are repeated on the deformed core geometry provided by the structural mechanics simulation. SHARP is uniquely posed to perform this simulation, as PROTEUS is the only deterministic neutronics code with an unstructured grid capable of solving the neutron transport equation on a highly complex deformed geometry. The complexities of mesh mapping and motion, solution transfer, and parallel efficiency are well suited for SHARP’s sophisticated integration tools.

Therefore, the effort reported here has two objectives: (1) to develop and demonstrate the capability to model core radial expansion phenomena with SHARP and (2) to evaluate the feasibility of modeling this phenomena for large-scale full-core applications. Such large-scale structural deformations have never been performed with SHARP, and some code development efforts were required to deform the geometry, smoothen the discretized mesh, and communicate these changes in a fashion suitable for the three physics modules. Therefore, to accomplish the first goal, a small 7-assembly mini-core model was developed to test the code integration capabilities that were developed. The fully coupled modeling capability was demonstrated for this test model.

To address the concern of feasibility of engineering-scale full-core simulations, a model of the Advanced Burner Test Reactor (ABTR), with 199 assemblies, was developed for each of the three physics modules. After surveying several options, the ABTR was chosen as the target demonstration problem for the following reasons: the design of ABTR incorporates structural mechanical feedback by assembly bowing, and information on the ABTR is readily available, unlike other facilities like the Fast Flux Test Facility. Results and evaluations of the computational effort associated with the ABTR analysis are provided in this report. The technology for performing the ABTR model has already been developed, and the engineering-scale coupled simulations can potentially be completed in the near future.

The 7-assembly mini-core and ABTR full-core test problems are described in the following section. This is followed by an overview of SHARP and a description of the code development efforts that were required to perform the coupled simulations. Mesh generation for each of the three physics codes is described in section 4. This is followed by descriptions of the simulations performed by the three individual physics modules (section 5), and a description of the coupled simulations (section 6).

2 Problem Specifications

Detailed specification of the ABTR problem is provided in a separate companion report [6], and therefore is only briefly summarized here. The 7-assembly problem is derived from the ABTR model, and thus will be described briefly in this section.

The ABTR is a conceptual advanced sodium-cooled fast reactor (SFR) designed by Argonne to serve as a prototype capable of demonstrating the technology of burning high-actinide fuel while simultaneously producing electricity [6]. ABTR is rated for a thermal power of 250 MW with an electric output of approximately 95 MW. As in most SFR designs, fuel subassemblies consist of an array of pins supported within a thin-wall hexagonal duct (Figure 2.1). The assemblies, which have a hexagonal cross section, are inserted in a lattice (Figure 2). The 199 assemblies can be categorized and counted as 54 driver fuel assemblies, 78 reflector assemblies, 48 shield assemblies, 10 control rod assemblies, 6 fuel test assemblies, and 3 material test assemblies. The 54 driver fuel assemblies are categorized into 24 inner zone driver assemblies and 30 outer zone driver assemblies. The inner zone driver assemblies have lower TRU enrichment (16.5%) than the outer zone assembly (20.7%), which helps to maintain a flattened power distribution. All ABTR assemblies have the same HT-9 hexagonal duct structure, SS-316 lower structure, and upper handling socket. Sodium flows through the gaps between assemblies.

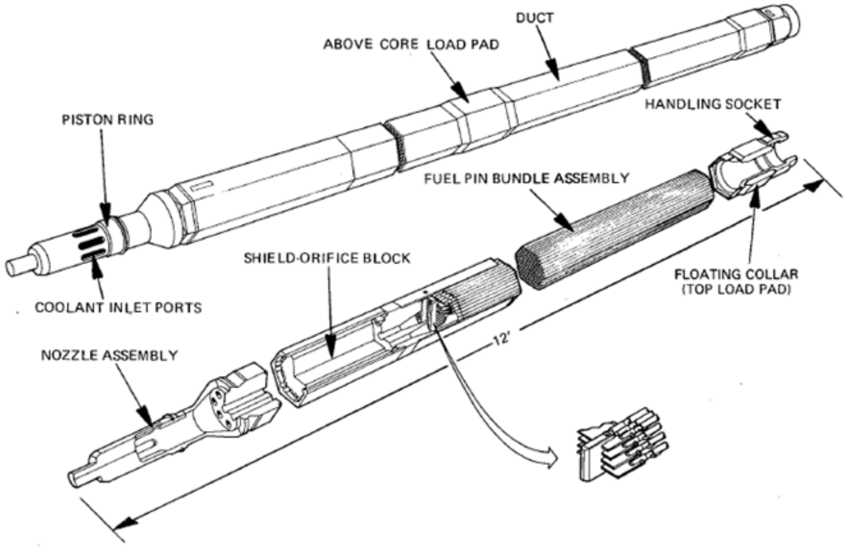


Figure 2.1. Typical SFR fuel assembly.

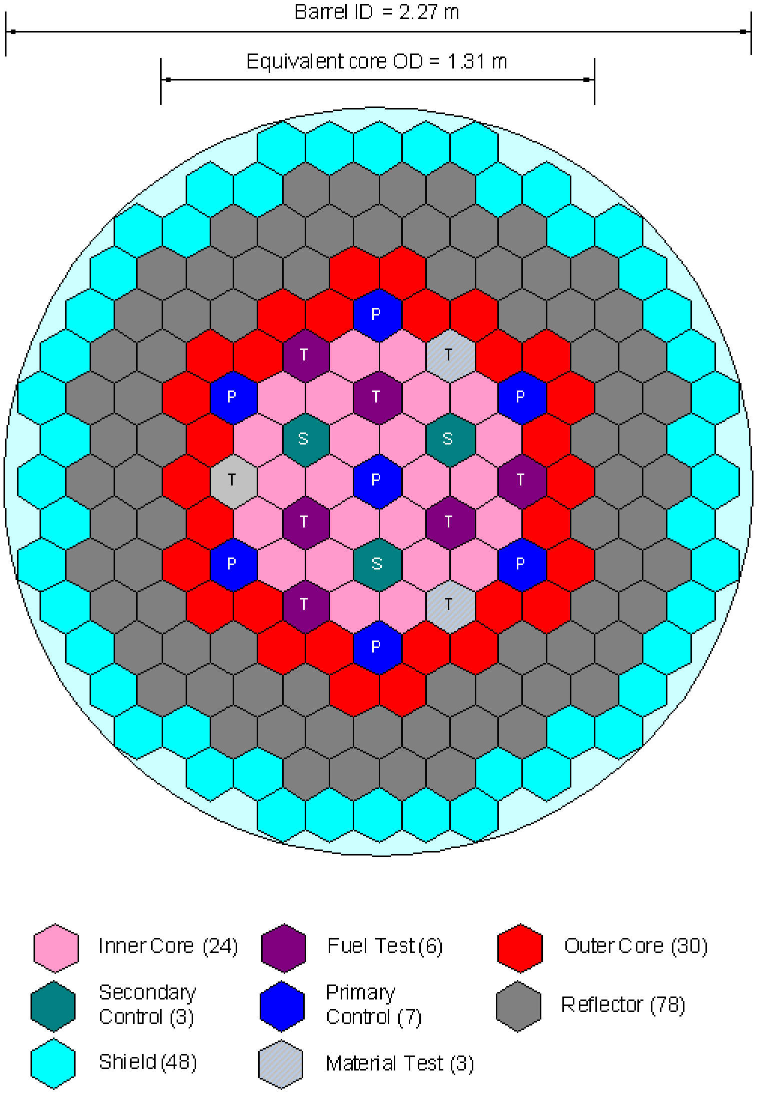


Figure 2.2. ABTR full-core assembly layout.

To ensure the negative reactivity response associated with bowing of the fuel assemblies, the ABTR utilizes the “limited free bow” core restraint system (Figure 2.3). The restraint system is characterized by top load pads (TLPs) on the assembly ducts at the top and above-core load pads (ACLPs) in the region above the core, along with restraining rings at the TLP and ACLP axial heights. The rigid restraint rings are attached to the core barrel at the ACLP and TLP locations. The load pads serve as preferential contact points between the ducts. The pads add only marginal thickness to the main duct body (thickness is exaggerated for clarity in the figure) but they are nonetheless thick enough to maintain the desired form under the design loadings. Additionally, the design ensures that duct-to-duct loading (resulting from bowed ducts in contact) is kept within allowable limits, including the time-dependent inelastic bowing effects due to irradiation (and thermal) creep and swelling effects.

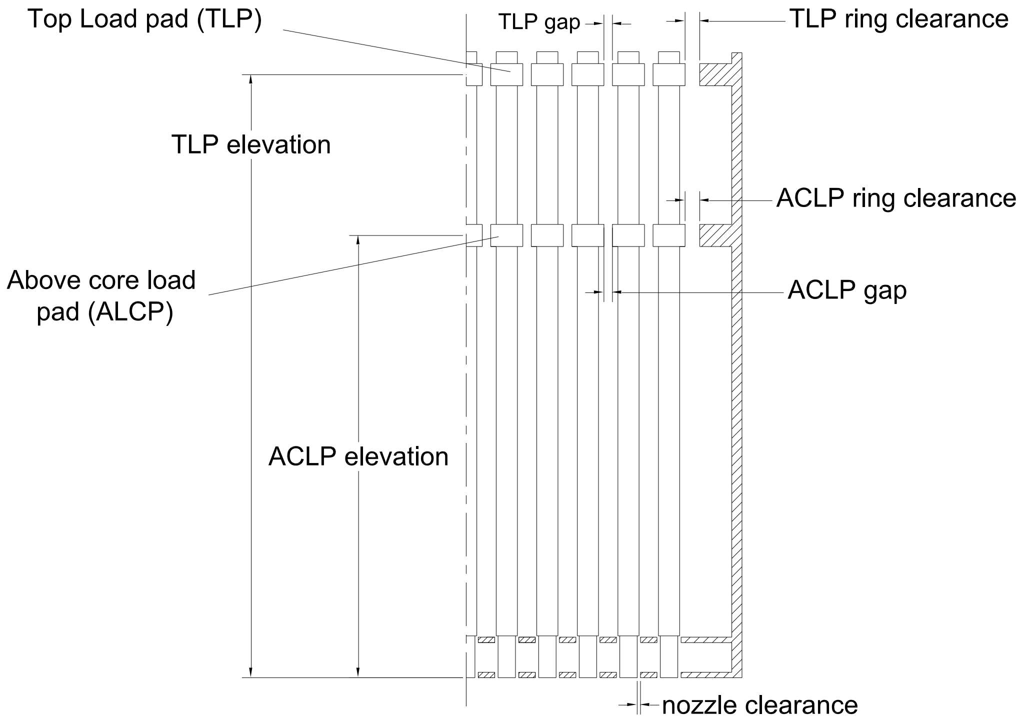


Figure 2.3. Limited free bow core restraint system.

The limited free bow core restraint system is designed to provide inherent protection against over power events by taking advantage of thermally induced bending action of the fuel ducts. This is illustrated in Figure 2.4, which shows a row of three cantilevered ducts located symmetrically about the center of a core and in a radially varying thermal gradient. Figure 2.4a shows the nominal configuration of the ducts with no temperature gradient. As the radial thermal gradient develops (temperature increases as distance from centerline decreases), the ducts begin to bow outward as shown in Figure 2.4b. Prior to contact with the top core restraint ring, the duct bends away from the core centerline as the temperature increases and therefore reduces the reactivity insertion. After contacting the top restraint ring and as the temperature gradient increases, the center of the duct bows inward which temporarily increases the reactivity. As the gradient increases, the inward bowing continues until the ducts contact at the ACLP. When the interior ducts all contact at the ACLP, the reactor is ‘locked-up’ and no further compaction can occur. Subsequent increased thermal gradients cause a reverse bowing below the ACLP moving the core region away from the core center as illustrated in Figure 2.4c. At this point, the reactivity generally decreases with constant negative slope as temperature increases. The core restraint system is designed to have this lock-up occur below the nominal operating core outlet temperature. In this way, the core is already locked up during normal operation, and any transient events with increasing temperature will induce further outward bowing in the middle of the core.

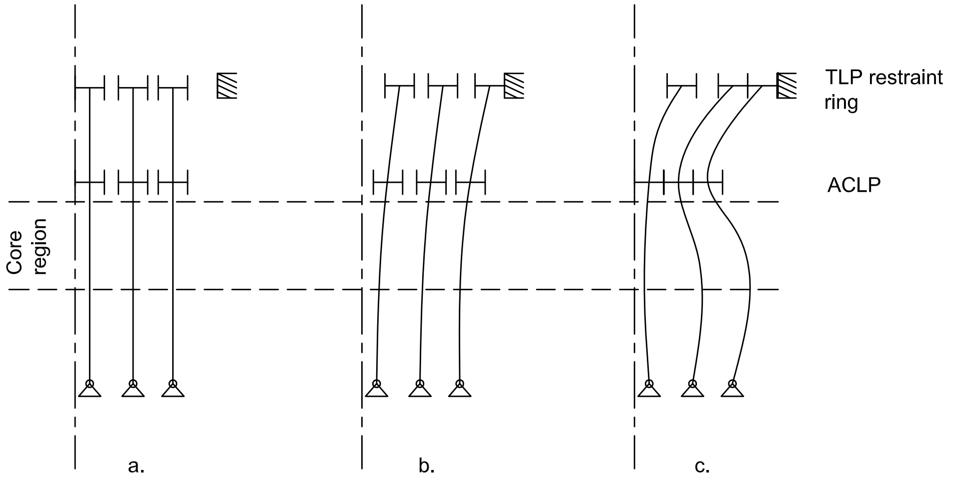


Figure 2.4. Operating mechanism of the limited free bow core restraint.

Multi-physics simulations of the core bowing phenomenon—which explicitly represent the duct wall motion and its influence on the temperature and neutronics fields—have been performed for a smaller test core with 7 assemblies, and physics models have been developed for the ABTR model described in the specification document [6]. Two versions of the 7-assembly problem were created: one that employs each of the different types of assemblies, and another that is comprised entirely of fuel assemblies. Both are shown in Figure 2.5. In both models, the fuel assembly is the "inner" ABTR core assembly shown in Figure 2.2. The model with three assembly types was designed to assess the models created for each assembly and ensure that the software being developed could handle models with multiple assembly types. The layout is known to be inconsistent with typical SFR designs, and would in fact produce undesirable power profiles. The all-fuel layout provides more realistic power profiles for the inner core region.

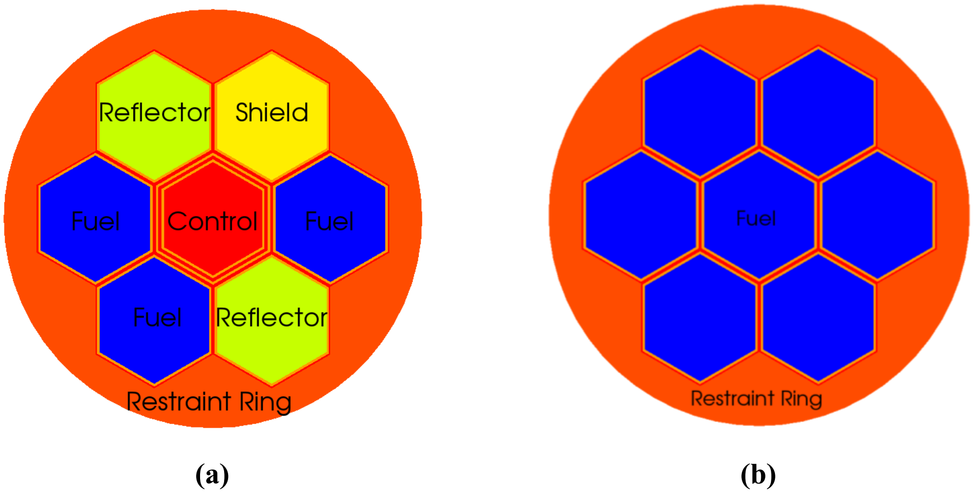


Figure 2.5. Seven-assembly mini-core models: (a) Four assembly types; (b) Only fuel assemblies.

3 Overview of SHARP and the Coupling Methodology

The NEAMS Reactor Product Line (RPL) aims to develop an integrated multi-physics simulation with a multi-resolution hierarchy that is designed to ultimately span the full range of length and time scales present in relevant reactor design and safety analyses, as well as scale from desktop to petaflop computing platforms. This section discusses the design and the numerical methodologies used in the SHARP toolkit to integrate neutronics, thermal-hydraulics, and structural mechanics physics components to perform coupled reactor analysis on a representative SFR core geometry. Based on the requirements specified, a problem to quantify the primary structural mechanical feedback effect with multi-way coupling has been implemented with dual resolution: a detailed heterogeneous model represents the duct surrounding each assembly while interior of the ducts (the individual assemblies) are represented with a homogenized geometry.

In order to produce a fully coupled-physics simulation capability, two obvious approaches can be pursued. In one approach, existing single-physics codes/components can be assembled into an overall coupled simulation code with appropriate interfaces to communicate between the components to capture the nonlinear feedback effects. This is generally referred to as a “small-f” or “bottom-up” framework approach [1, 8]. The other approach is to use an integrated, coupled-physics modeling framework, with new code pieces for each relevant physics area developed inside that framework from scratch. This is sometimes referred to as a “large-F” or “top-down” approach [9, 10]. The primary advantage of the former approach is that it preserves several man-years invested in existing verified and validated individual physics modeling codes, but at the cost of some intrusive modifications to enable the software interfaces. The large-F approach avoids intrusive interfacing by providing a unified platform to enable coupling, but at the cost of re-writing all the necessary physics codes and verifying the components individually and as a whole. The overall approach being pursued in the RPL effort is to develop and demonstrate a small-f framework for performing coupled multiphysics analysis of reactor core systems. This system takes advantage of many single-physics codes also sponsored by the overall NEAMS program over past several years.

This relevant detail regarding the background on construction of the RPL coupled-physics framework (SHARP) along with the methodology is discussed in the following sections.

3.1 The SHARP Multi-physics Code System

A multi-physics reactor core modeling code can be constructed in many ways, and numerous past efforts have provided stepping-stones for future efforts [10]. What distinguishes the SHARP effort from others is the goal of flexibility in the physics, discretization types, and software options supported by the framework. This section describes the SHARP modeling approach in detail and illustrates how various existing physics codes have been connected to this framework.

As stated above, SHARP employs a “bottom-up” approach, so it can use existing physics codes and take advantage of existing infrastructure capabilities in the MOAB framework and the coupling driver/solver library, the Coupled Physics Environment (CouPE), which utilizes the widely used, scalable PETSc library [11].

Using an existing physics code in this system (Fig. 3.1) requires that the system support the mesh type used by the individual physics models. The physics models can retain their own native representation of the mesh, which gets transferred to and from MOAB's representation through a mesh adaptor; or it can use MOAB's representation directly. Language interoperability through the C/Fortran-based iMesh interfaces also allows flexibility in the implementations that are tuned to individual physics requirements without overhead.

In practice, this means that the coupled system may be solved on multiple meshes, each of which models part or all of the physical domain of the problem. To perform efficient coupled calculations, the results must be transferred from the mesh on which they are generated (source mesh), to the mesh for which they provide initial or boundary conditions (target mesh) due to nonlinearity introduced because of coupling between physics models. "Multi-way" transfer is required in cases where the physics depend on each other's solution fields, for example in reactor analysis where neutronics computes heat generation based on temperature properties computed by thermal-hydraulics, which in turn depends on the heat-generation source term computed by neutronics.

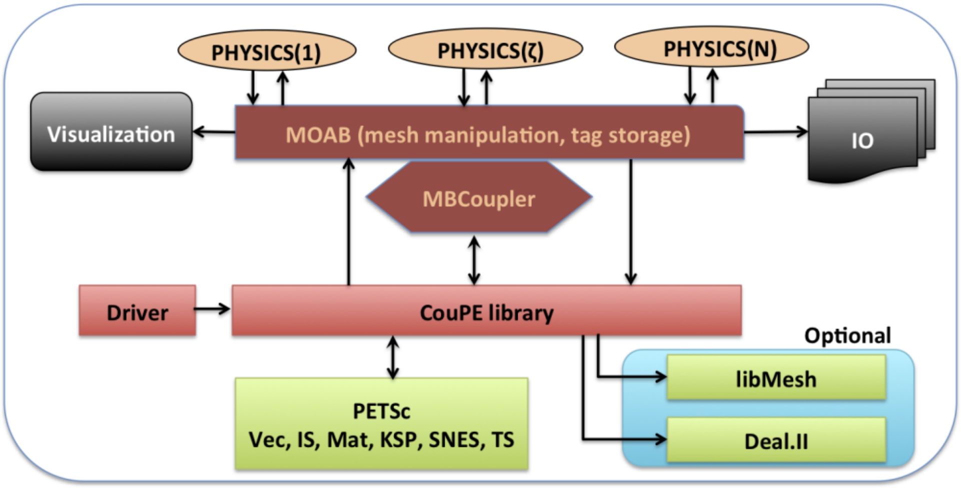


Figure 3.1. SHARP architecture.

Since relevant physics components solving a nuclear engineering problem have widely varying backgrounds in terms of code architectures, dependency requirements, and specialized solver data-structures, a flexible approach to the coupling methodology was necessary to obtain accurate solutions. This motivation led to the development of the MOAB-based spatial projection tools and the CouPE drivers based on PETSc library to orchestrate the global nonlinear solver. Details regarding these tools are given in the following sections.

3.1.1 Mesh Database (MOAB)

One of the critical aspects in assembling a multi-physics modeling code is mapping the results from one physics domain to another. In the small-f RPL framework, a common mesh library serves this purpose. The MOAB library provides a "data backplane" to link physics models through their spatial domains, and MOAB's MBCoupler package transfers physics results between those domains.

MOAB is a generic library for query and modification of structured and unstructured mesh and field data associated with the mesh [2]. MOAB can represent all entities typically found in the finite-element zoo, as well as polygons, polyhedra, and structured meshes. MOAB provides parallel functionality for resolving entity sharing and ghosting between processors, with sharing and ghosting information available as annotations on the local mesh. MOAB's parallel I/O is based on the parallel HDF5 library, and it has been demonstrated on processor counts up to 16,000 (on the IBM BlueGene/P system). A partitioning tool has been implemented by interfacing with the Zoltan partitioning library implementation, with in-situ visualization provided by a Paraview plugin. MOAB can read meshes generated by the CUBIT mesh generation toolkit and can represent the various mesh types used in this effort.

MOAB's data model consists of four fundamental data types:

- Entity: A basic entity in the discrete model, e.g., vertex, quadrilateral, tetrahedron.
- Entity Set: An arbitrary collection of entities.
- Tag: A piece of data annotated on entities and sets.
- Interface: The primary database object instance.

Even though this data model seems simple, it can represent all the necessary data to run coupled simulations. In particular, tags can be used to store both fine-grained solution data on individual vertices and elements and coarse-grained annotation of sets to identify them as boundary conditions, material types, or processor partitions. Two particular groupings are common to this effort:

- Material sets, also referred to as "element blocks," group elements by material definition. In MOAB, these are represented as entity sets, tagged with a "MATERIAL SET" tag whose value stores a user-assigned id number.
- Neumann sets, also referred to as "sidesets," store groups of lower-dimensional entities (in a 3D mesh, Neumann sets contain mesh faces, for example). Similarly, these groups are marked with "NEUMANN SET" tag whose value is the user-assigned id number.

Other tags used to store field and other data for PROTEUS and Nek5000 are described in sections 3.2 and 3.3, respectively.

3.1.2 Solution Transfer Tool (MBCoupler)

MBCoupler, a MOAB-based tool for solution transfer [12], has been demonstrated on up to 4,000 processors. This tool allows the source and target meshes to be distributed across processors in whichever way is best suited for the physics associated with each mesh. Target-to-source mesh point location is performed in parallel, with bounding-box-based acceleration used to determine possible source mesh processors containing every point and with KD-Tree decomposition used locally on each processor. This tool can transfer solutions using both linear finite element and piecewise-constant shape functions. As described in Section 3.4, it has recently been extended to incorporate spectral element shape functions as well. In the demonstration described in this report, MBCoupler is used to map the results computed by one physics module to boundary conditions on the mesh used by the next physics module.

3.1.3 Coupled Physics Environment (CouPE)

CouPE provides scalable and extensible interfaces to couple different physics components that are nonlinearly dependent on each other. The SHARP multiphysics coupled code for reactor analysis problems employs validated and verified efficient single-physics codes with message passing interface architecture to achieve tight coupling with an iterative operator split methodology [10, 13]. Such iterative nonlinear methods provide the flexibility to use standard industrial codes and avoid replicating man-years of development and testing by following the bottom-up approach (section 3.1).

The aims in designing the CouPE code library included the following:

1. Make use of existing libraries and physics codes to minimize development time and base the framework on already well-verified and validated single-physics codes and libraries.
2. Enable a flexible and accurate data exchange framework between codes in a mesh, numerics, and physics aware fashion, i.e., maintain consistency, accuracy, and conservation of key fields.
3. Provide flexible data containers and physics objects to facilitate and simplify the evaluation of the non-linear residuals representing the fully discrete partial differential equations for different physics components.
4. Employ different kinds of multi-physics coupling strategies within the same architecture with minimal changes in the driver.
5. Enable runtime object polymorphism.

CouPE aims to solve all of the physics components under a unified framework in order to exchange the solution from one physics model to another and converge the coupled-physics solution fields to user-specified tolerances without sacrificing numerical stability or accuracy. CouPE provides the necessary components and layers to wrap existing physics codes or write a complete description of a physics problem from scratch to solve phenomena of interest, that is to enable both bottom-up and top-down approaches. The library also provides the necessary tools to quickly implement any of the popular variations of an operator-split coupled solver (Marchuk, Strang, Yanenko among others [13]) or a more rigorous matrix-free inexact-Newton solver with a Jacobian-free Newton-Krylov (JFNK) technique [14]. Currently, the Marchuk splitting with Picard iteration over the physics components has been implemented in CouPE; other coupling strategies have been implemented but need to be tested for relevant multi-physics problems.

For stationary, coupled nonlinear problems, the primary source of error stems from the exchange of physics solutions that reside in different spatial discretizations and resolutions. CouPE utilizes the iMesh interfaces. And, more specifically, its implementation by MOAB and MBCoupler enable seamless integration of the single-physics codes. This is made possible by exposing a minimal interface to be implemented by the physics wrappers, whose design follows the software paradigms of PETSc [15]. The current design of CouPE is intended to satisfy the need for a loosely coupled software framework to solve strongly coupled physics modules. The implementation of coupled methods is usually difficult and CouPE can reduce the development time by providing a template to solve a collection of nonlinearly coupled physics objects via a uniform interface. The driver is simple, transparent, and extensible. It can be thought of as a "glass-box" solver rather than a "black-box" solver, since it provides access to all the internal details of the physics and the corresponding internal mesh structures, and it allows the user to supply and override the behavior at runtime.

Similar to the PETSc toolkit library, CouPE is designed to allow the user to specify command-line arguments to control the dynamic behavior of the coupled solver. The parameter specifications include the input for individual physics components, input mesh parameters, and type of the solver, and in advanced usage, can even dynamically change the type of the physics being coupled. This is made possible by completely abstracting the behavior of the core object until runtime, even though the internals of these objects are fully available to the driver. Hence, the core implementation of a physics object is hidden while the driver utilizes only the methods exposed in the public interface. The advantage of such a method is that the implementation of the coupled physics driver and the accompanying physics components need to be compiled, linked, and verified only once and then can be reused in a variety of different coupling methods (e.g., loose versus tight coupling).

3.2 SHARP Physics Components

In the SHARP framework, MOAB interfaces are implemented for 3 different physics components that are relevant to fast reactor physics analysis. The addition of a new physics component to the framework requires integration and ability to read the mesh and possibly associated data from iMesh/MOAB formats, along with implementation to propagate solution variables back onto the mesh after their computation via tags defined either on discrete vertices or elements. Because of the various storage formats used in physics models, and the parallel domain-decomposed environment in which these calculations are usually run, this integration process can be somewhat involved.

To better understand the level of fidelity that can be achieved by the SHARP framework, some key aspects of the 3 physics components are given below.

3.2.1 Neutron Transport Solver (PROTEUS)

PROTEUS is a high-fidelity deterministic neutron transport code based on the second-order even-parity formulation [16]. The application scope targeted for PROTEUS ranges from the homogenized assembly approaches prevalent in current reactor analysis methodologies to explicit geometry approaches, with the ability to perform coupled calculations to thermal-hydraulics and structural mechanics. The PROTEUS solver has a proven capability of using existing petascale parallel machines to solve problems with demonstrated scalability of over 70% (strong scaling) at over 250,000 processors (on BlueGene/P). These achievements of PROTEUS were made possible by partitioning the space-angle system of equations over the available processors and utilizing established iterative solution techniques from the neutron transport community combined with the parallel algorithms in the PETSc toolbox.

Interfaces to the MOAB mesh database have been written to handle UNIC meshes that describe detailed geometries with multiple blocks (regions) with appropriate specification hooks for temperature- and density-dependent material cross-section evaluation and interpolation. This interface is essential to capture the nonlinear feedback effect from thermal-hydraulics. Inherent ability to use a deformed mesh with appropriate recalculation of the density changes within materials (thereby affecting cross-sections) have also been implemented to enable direct coupling to a deformation code such as Diablo.

The eigenvalue solver in PROTEUS computes the neutron flux shape, computes the power distribution in the reactor, and then places the computed data in appropriate MOAB tags. The power solution field is then propagated to the other physics solvers via the data-coupling interfaces that support tight coupling with thermal-hydraulics, which uses the tag data as a thermal source term to compute temperature fields. Several verification studies have been performed during the quality assurance process to ensure that the coupled solver solutions are physically meaningful.

3.2.2 Computational Fluid and Thermal Dynamics Solver (Nek5000)

The Nek5000 computational fluid dynamics solvers are based on the spectral element method developed by Patera [17]. Nek5000 supports two different formulations for spatial and temporal discretization of the Navier-Stokes equations. The first is the P_{N}-P_{N-2} method with velocity/pressure spaces based on tensor-product polynomials of degree *N* and *N-2* respectively. The second is the low-Mach number formulation of Tomboulides and Orszag [18], which uses consistent order-N approximation spaces for both the velocity and pressure. The low-Mach number formulation is also valid at the zero-Mach (incompressible) limit [19]. The Nek5000 code has been extensively verified and validated for several benchmark problems and has a proven scalability in existing petascale architectures up to 131,072 processors (over a billion degrees-of-freedom).

The conjugate heat transfer problems that are typically present in nuclear engineering applications can be solved rigorously using the formulations in Nek5000. Typically, the following boundary conditions are applied at the inflow, outflow, and wall surfaces:

- The inlet surface has uniform prescribed velocity and fixed temperature,
- The outlet surface has standard outflow boundary conditions, and
- The wall surfaces have velocity non-slip and insulated temperature boundary conditions.

Using the standard iMesh-based interfaces to MOAB, several different mesh formats can be natively used with Nek5000 along with the extended ability to couple with other physics components in the SHARP framework. When running Nek5000 in the fully coupled mode, the fluid/solid temperatures, along with their corresponding densities, are stored in MOAB tags to be used either by the structural mechanics or neutronics components, propagated by the unified data-transfer mechanisms detailed earlier. The MOAB interface also enables Nek5000 to perform application checkpointing and have restart capabilities independent of the number of processors, thereby enabling the user an opportunity to investigate the validity of the coupled solution before proceeding further.

In order to perform thermofluid analyses of homogenized fuel assemblies, a porous media model was implemented in the Nek5000 code. The porous media model, sometimes referred to as a distributed resistance model, is based on the model implemented into the STAR-CD code [16]. Porous media models are typically applied to problems where the fluid flows through a region with many small-scale solid structures, and it would be impractical to resolve the geometry explicitly. Instead, the effect of the small-scale solid structures on the flow is modeled as a momentum sink or resistance in a homogenized fluid domain. In this particular case, we wish to model the influence of the fuel pins on the flow, i.e., drag and pressure drop, as a momentum sink without explicitly representing the geometry of thousands of fuel pins. The model must also account for the energy deposition associated with nuclear fission. Moreover, fuel and cladding temperatures are estimated for each fuel assembly, and may be provided to the neutronics code in future coupled simulations.

Because the porous media model employs a fairly standard model, which may be found in the STAR-CD manuals [20] among other sources, it is only summarized here. First, a volume porosity χ is defined as the ratio of open volume to total volume of the porous medium. This is used in all time-derivative terms in the mass, momentum, and energy continuity equations to provide the appropriate fluid inertia.

An additional body force per unit volume is added to the momentum equation such that:

$$\boldsymbol{F}_p = -K \cdot \mathbf{v} \qquad (1)$$

where K is the porous resistance tensor and $\mathbf{v}$ is the superficial velocity. Superficial velocity is defined as the volumetric flow rate divided by the total cross-sectional area. For each of the three directions (i = 1,2,3), K is a diagonal matrix given by:

$$K_{ii} = \alpha_i \lfloor \mathbf{v} \rfloor + \beta_i \qquad (2)$$

where α_i and β_i are model-dependent coefficients with dimensions of [mass × length^{-4}] and [mass × length^{-3} × time^{-1}], respectively. Note that the repeated indices denote the diagonal elements of the tensor, not summation. For channel flow, the resistance may be considered orthotropic, i.e., only causing resistance in the Z-direction. An appropriate choice of α and β can be determined from an empirical formulation:

$$\alpha = \frac{PC_1 \rho Re^n}{2\chi^3 A} \qquad (3)$$

$$\beta = \frac{PC_1 C_2 \mu Re^n}{2\chi^2 A D_h}$$

where C_1, C_2, and n are model coefficients; P is the wetted perimeter; A is the superficial area; D_h is the hydraulic diameter; ρ is the density; and μ is the molecular viscosity.

The influence of turbulence on the momentum transport is assumed to be included in the porous resistance term in Eq. (1). However, turbulence is included in the diffusional flux term in the energy transport equation in the fluid, which is conventionally:

$$F_{h_j} = -\lambda \frac{\partial T}{\partial x_j} + \bar{\rho}\overline{u_j' h'} \tag{4}$$

where λ is the thermal conductivity, T is the temperature, u_j' is the fluctuating component of velocity in the x_j-direction, and h' is the fluctuating component of enthalpy. The second term in this expression represents turbulent diffusion of thermal energy. This term is evaluated using the turbulent kinetic energy k and dissipation ϵ from the k-ϵ turbulence model:

$$\begin{gathered} k = \frac{3}{2} I^2 \lfloor \mathbf{v} \rfloor^2 \\ \epsilon = \frac{C_\mu^{\frac{3}{4}} k^{\frac{3}{2}}}{L} \end{gathered} \tag{5}$$

where I is the turbulence intensity, L is the turbulence length scale, and C_μ is a coefficient that equals 0.09 in the standard k-ϵ model. Each of these is strongly dependent on the model. These values may then be used to evaluate the turbulence viscosity ν_t using the following relation:

$$\nu_t = \frac{C_\mu k^2}{\epsilon} \tag{6}$$

Then given a value of the turbulent Prandtl number Pr_t, the thermal diffusivity term can be evaluated.

Heat is generated by nuclear fission in the fuel, conducted through the fuel and cladding, and removed by the coolant. The fuel and cladding are both part of the solid portion of the domain, which is causing the flow resistance. An interphase heat transfer term is added to the coolant energy transport equation, which requires the cladding outer surface temperature $T_{clad,o}$.

$$h_x\left(T_f - T_{coolant}\right) \tag{7}$$

where h_x is the thermal conductance per unit volume and $T_{coolant}$ is the coolant temperature of the permeating fluid. Energy transport equations for the solid components must also be written, with volumetric heat generation in the fuel and energy transfer from the fuel to the coolant that matches Eq. (7). The equations are solved in a coupled fashion..

3.2.3 Solid Mechanics Solver (Diablo)

The Diablo code being developed at Lawrence Livermore National Laboratory uses implicit, Lagrangian finite-element methods for the simulation of solid mechanics and multi-physics events over moderate to long time frames [5]. A primary focus is nonlinear structural mechanics and heat transfer. The code provides a venue for applying parallel computation to discretization technologies developed and user-tested in the legacy serial-processor codes NIKE3D and TOPAZ3D. Diablo is built around Fortran 95 data structure objects and a message-passing programming model. The architecture provides flexibility for the addition of other field problems, such as electromagnetics.

In structural analysis of mechanical assemblies, a key functionality is "contact": capturing the interaction between unbonded material interfaces. The Diablo team has broad experience with contact problems and has created state-of-the-art algorithms for their solution. Their experience with contact motivates the use of low-order spatial discretizations, such as eight-node hexahedra for continua and four-node quadrilaterals for shells. Appropriate formulations are employed to accommodate nearly incompressible material models, such as for metal plasticity and rubber elasticity. Global algorithms include second-order and quasi-steady time integration and a number of approaches for nonlinear iteration: full Newton, modified-Newton, multiple quasi-Newton updates, and line search. Linear solvers are utilized from multiple libraries.

3.3 Multi-Physics Coupling Methodology

In the future, Diablo, PROTEUS, and Nek5000 will all run simultaneously underneath the CouPE framework and communicate quantities through MBCoupler in MOAB. As an interim step, the coupling has been accomplished through file-based transfer. This is a 2-step process, with the second step consisting of 8 substeps.

1. Individual Nek5000, PROTEUS, and Diablo meshes are generated in the undeformed configuration. Nek5000 and PROTEUS use MOAB mesh files natively. Currently Diablo uses an EXODUS input file and writes the equivalent MOAB (".h5m") file as part of the initialization process. Thus, four mesh files are prepared:
    a. NEK.in.h5m
    b. PROTEUS.in.h5m
    c. DIABLO.in.exo
    d. DIABLO.in.h5m (created by Diablio when it initializes)
2. SHARP iterates the problem until convergence:
    a. Coupled 2-mechanics runs (PROTEUS and Nek5000) are made using the updated mesh
    b. Temperature data from Nek5000 is written to its native "FLD" file format, NEK.temps.FLD

c. The VisIt utility converts the "FLD" file format to a MOAB ("h5m") file, NEK.temps.h5m

d. A standalone version of MBcoupler maps the Nek5000 data in .h5m format to the Diablo .h5m file, DIABLO.temps.h5m

e. Diablo uses the temperature data and coupled solid mechanics to produce deformations, which are written as scalar quantities UX, UY, and UZ to an undeformed MOAB database, DIABLO.disp.h5m

f. The standalone version of MBCoupler maps the UX,UY,UZ data to the Nek5000 and PROTEUS meshes,
   i. NEK.disp.h5m
   ii. PROTEUS.disp.h5m

g. The standalone utility DEFORM moves the vertex coordinates on the PROTEUS and UNIQ meshes according to the mapped values of UX, UY, and UZ,
   i. NEK.deformed.h5m
   ii. PROTEUS.deformed.h5m

h. PROTEUS densities and isotope volume fractions are updated based on the mesh deformation if so desired.

i. The deformed meshes are used as inputs to repeat step 2a above and continue the iterations, as depicted in Fig. 3.2.

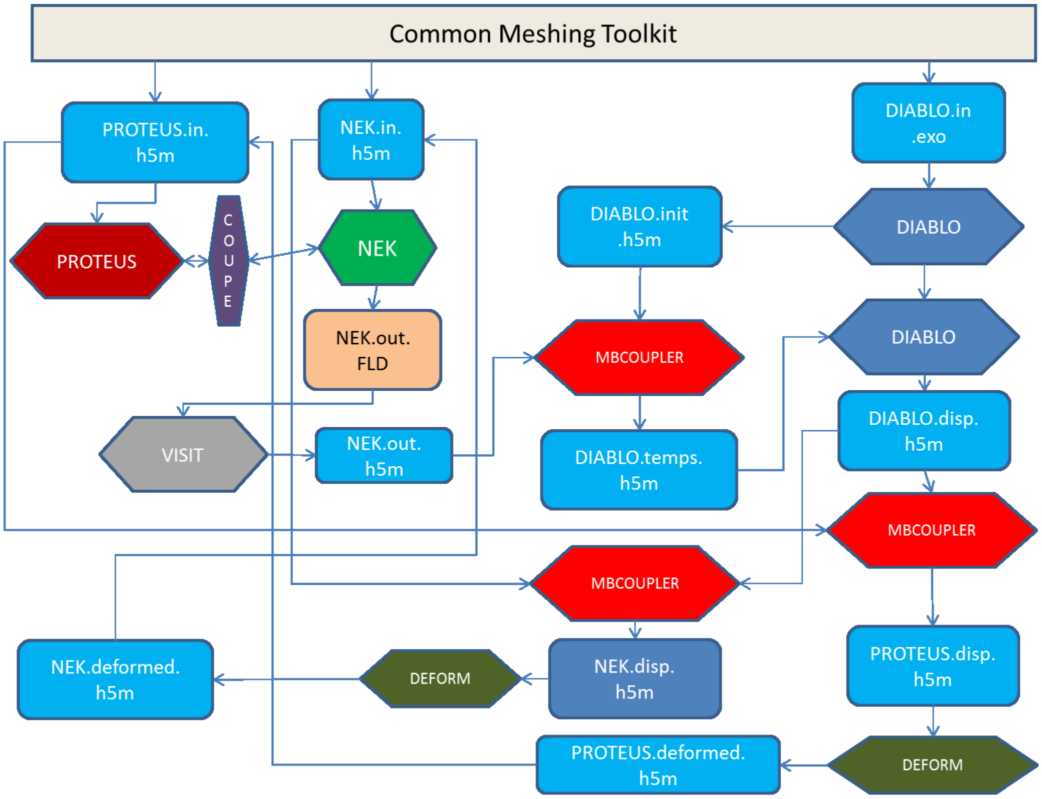


Figure 3.2. Coupling and iteration process.

4 Mesh Generation

This demonstration used the reactor geometry (and mesh) generation (RGG) tools in MeshKit for creating the meshed core models. The RGG tools use three-stage methodology comprising three modules –AssyGen, Meshing, and CoreGen. Details of methodology, literature review, handling memory, automation, parallelism, and various results are presented in other papers [21–23] and reports [24–26]. The two key algorithms are contained in AssyGen and CoreGen:

1. AssyGen generates assembly geometries and journal files for meshing of hexagonal and rectangular assembly lattices based on a text-based input file.

2. CoreGen reads an input file describing the reactor core arrangement and generates the reactor core mesh or geometry from its component mesh or geometry files, respectively. It inserts the assemblies into the overall core model, and then merges the matching nodes at the interfaces of assemblies and interstices mesh to form the whole model.

Details on key modeling assumptions, geometry creation, meshing process and salient features of this methodology are presented in the following sub-sections.

4.1 Key Assumptions and Requirements

Apart from the modeling approximations described in section 3, several meshing restrictions and assumptions were made for modeling the ABTR assemblies, restraint-ring, and full core mesh. PROTEUS/Diablo and Nek5000 impose certain features and required characteristics on the mesh:

1. Geometry models are created using dimensions when the reactor is cold.
2. The model is homogenized; pins and other instrumentation inside the assemblies are not modeled.
3. All 199 assemblies are modeled with different material and boundary conditions in order to specifically identify and prescribe inlet/outlet boundary conditions to a particular assembly. Each assembly is numbered.
4. The axial and radial mesh size is coarse to keep the element count low.
5. The nosepiece region at the bottom of each assembly is not modeled.
6. Additional axial materials are created for modeling fuel regions in PROTEUS.
7. Nek5000's spectral element solver requires hex27 elements.
8. Nek5000 requires a plenum region that connects all the coolant flowing through the individual assemblies at the top of the reactor. The axial height of outlet plenum is 30cm.
9. Diablo and Nek5000 model fuel elements as a uniform "mush" across all the axial fuel regions.
10. PROTEUS needs only 3 boundary conditions, top, bottom, and side of the entire core.
11. Boundaries modeled for Nek5000 and Diablo are:
    a. Inner/outer walls of the restraint rings
    b. Wall of gap between assemblies and restraint rings
    c. Inlet for whole core and each assembly
    d. Outlet for whole core and each assembly
    e. Wall of TLP for each assembly
    f. Wall of ACLP for each assembly

4.2 Outer Covering / Restraint Ring Modeling

Outer covering and restraint rings are specified to have two gaps at axial regions of the TLP and ACLP. These gaps serve as a tolerance before reactor initiates fission; they deform, contact, and lock the assemblies as the simulation progresses. The restraint ring and outer core gap at ACLP is very small (0.0235 cm) and has to be modeled after creation of the restraint ring geometry. The mesh interval on the edges of the hexagonal ducts and along the height must match the assembly meshes described in Section 4.3. Five steps are required to create the restraint ring:

1. Use CoreGen to create the geometric model of the whole core; this is done by changing the “ProblemType” keyword to “Geometry” in CoreGen input file that describes the whole core.
2. Use AssyGen to create a cylinder with a radius of the outer covering and the same axial-subdivisions as the core model created in step 1.
3. Subtract the result of step 1 from step 2. See Figure 4.1.
4. Create gap geometry by gathering the hex curves on top or bottom and section the geometry to create the gaps shown in Figs.4.2 and 4. 3.
5. Mesh: First create a surface mesh at the top surface of ACLP, TLP and outer ring, then extrude the mesh and finally, prescribe the material and boundary conditions.

Creation of the interstices mesh (restraint ring/outer covering) is the only manual step in the core model creation. CUBIT was used to create this mesh. AssyGen automatically creates other component assemblies meshes (section 4.3).

4.3 Assembly Geometry and Mesh Creation (AssyGen)

All the homogenized assemblies are created using AssyGen. Twenty-four different axial regions are identified to satisfy the material requirements of all the assemblies and maintain the same axial interval on all assembly meshes. The range of these 24 axial regions is shown in Appendix A. New keywords “SaveExodus,” “NumSuperBlocks,” and “SuperBlocks” were introduced in AssyGen. “NumSuperBlocks” specifies the number of superblocks, and “SuperBlocks” specifies the blocks to be merged to form the new superblock. For example, PROTEUS models each fuel pin with tens of materials along the height, whereas thermal-hydraulics models consider all the fuel pins to be one material. These keywords help in creation of one-mesh files that can be used by all physics simulations. Material and boundary conditions that are not required by a particular physics are ignored. Note that superblocks are not required when modeling meshes for individual physics separately.

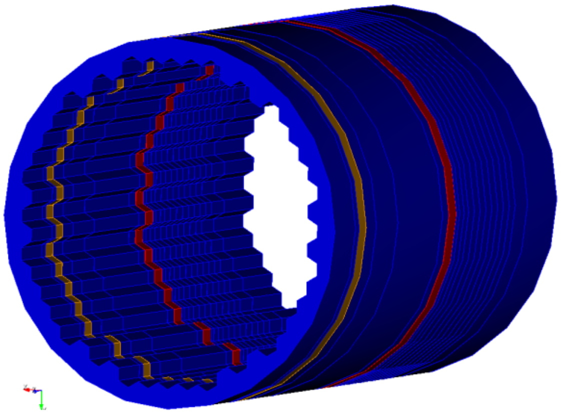


Figure 4.1. Geometry of restraint ring after subtraction of core geometry.

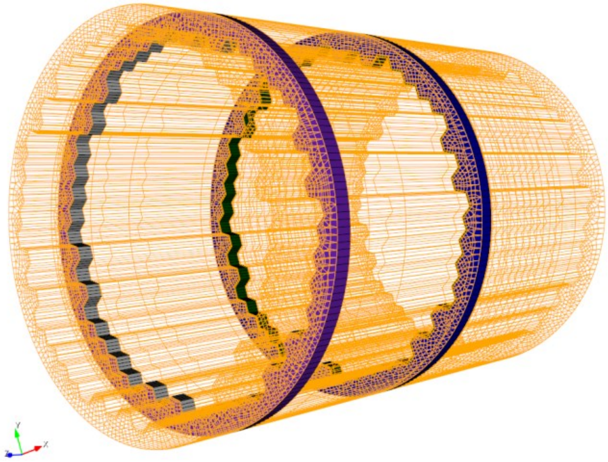


Figure 4.2. Mesh of restraint ring with gap at ACLP and TLP regions (Restraint ring is shown in purple, the gaps at ACLP and TLP are shown in gray and green respectively.)

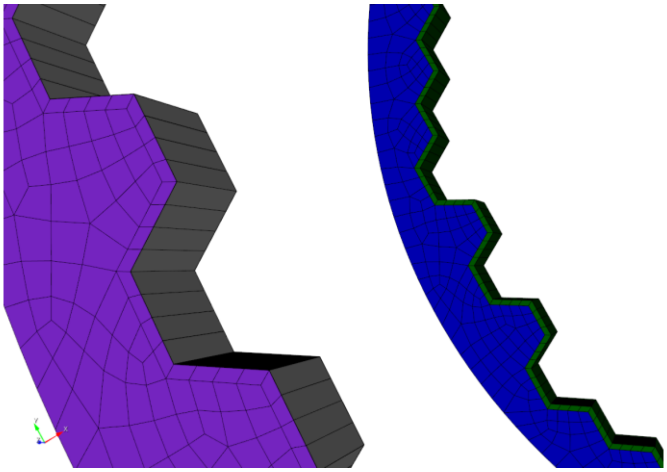


Figure 4.3. Close-up of restraint rings showing the gap at the ACLP (0.0235 cm, gray) and TLP (1.2025 cm, green).

4.3.1 Fuel, Inner Core, Outer Core, Shield, and Reflector Geometry

The geometry of homogenized fuel, inner core, outer core, shield, and reflector assemblies are the same. AssyGen files are created for each of these assemblies with different material definitions along the axial direction. Material properties along the axial height of each of these assemblies are described in Section 3. Details of modeling only the fuel assembly are presented in this section. AssyGen creates a geometry file and a mesh script for the homogenized assemblies shown in Fig. 4.4. See Appendix A for the AssyGen input file for this fuel assembly. Hex meshes are obtained by running CUBIT on the journal file created by AssyGen. The mesh journal file sets up the intervals on the edges, followed by meshing the top surfaces and extruding the surface mesh. All of the geometry creation and meshing steps have been automated in AssyGen. Each assembly has load pads, which are modeled by specifying separate material along the duct at the ACLP and TLP locations shown in Fig. 4.4b. Interassembly gap regions are divided in two equal parts and modeled with every assembly. All the interassembly gaps are merged when CoreGen assembles the individual assembly meshes to create the core model (as described in section 4.4.3).

The total wall-clock time required to create the homogenized geometry and mesh for this fuel assembly is about 2 minutes. The mesh has 2,500 hex elements. The detailed fuel assembly geometry model (nonhomogenized, Fig. 4.4a) with 217 fuel pins and varying material specifications contains about 15,000 geometric volumes. The dimensions of a single fuel pin are shown in Fig. 4.5; each pin for a particular axial height consists of four geometric volumes. It takes AssyGen about 20 minutes to create the nonhomogenized geometric model. The 217-pin nonhomogenized models were not used in the simulations described in this report.)

4.3.2 Control Assembly Geometry

The control assembly has an additional duct and sodium layer compared to the assemblies described in section 4.3.1. The AssyGen input file used for creating control assembly shown in Fig. 4.6b can be found in Appendix B.

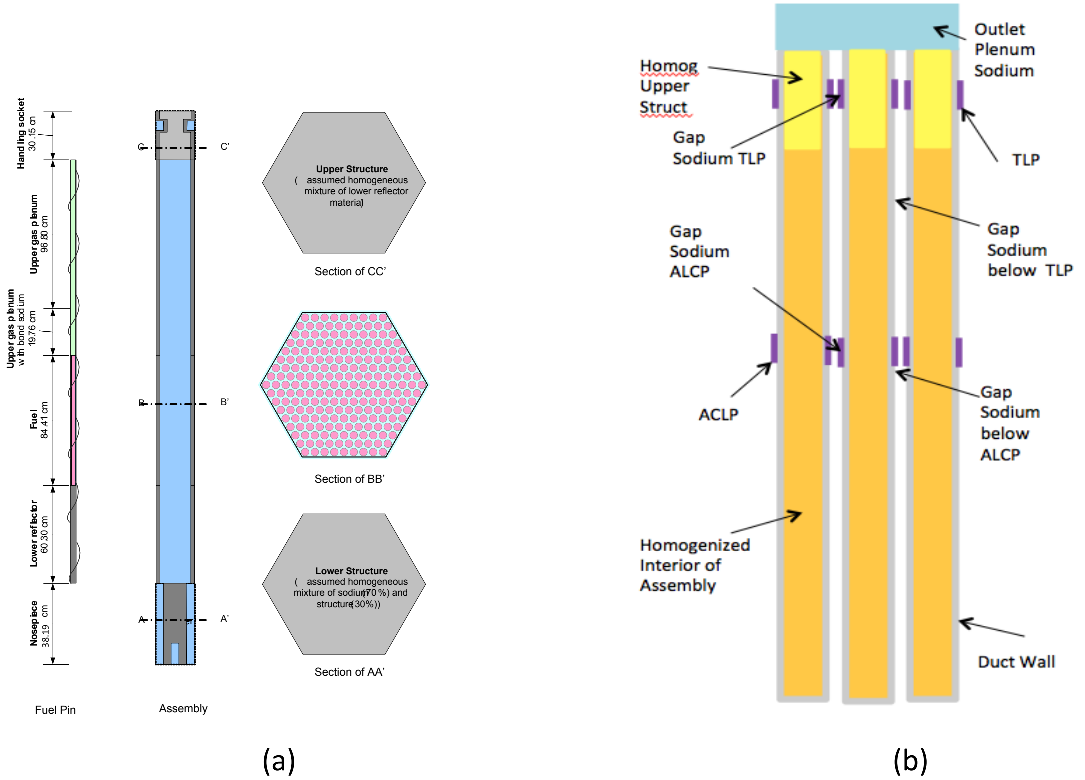


Figure 4.4. ABTR fuel assembly: (a) Detailed (nonhomogenized); (b) Three homogenized assemblies showing the ACLP, TLP, and outlet plenum regions.

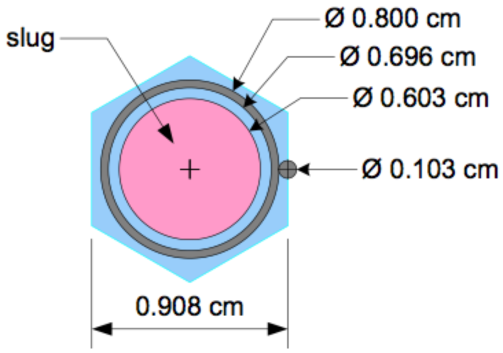


Figure 4.5. Section of metal fuel pin showing dimensions in cold condition.

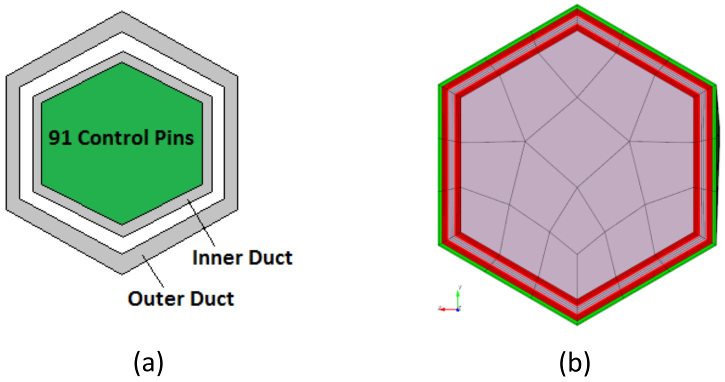


Figure 4.6. (a) Control assembly absorber region containing two ducts; (b) Inlet of actual control assembly with homogenized control pins (magenta), two ducts (red), and half of inter-assembly sodium gap (green).

4.4 Core Modeling (CoreGen)

This section highlights two mini-core models for testing standalone and coupled physics simulations. This discussion is followed by details on the full core model.

4.4.1 Mini-Core with Four Different Assemblies

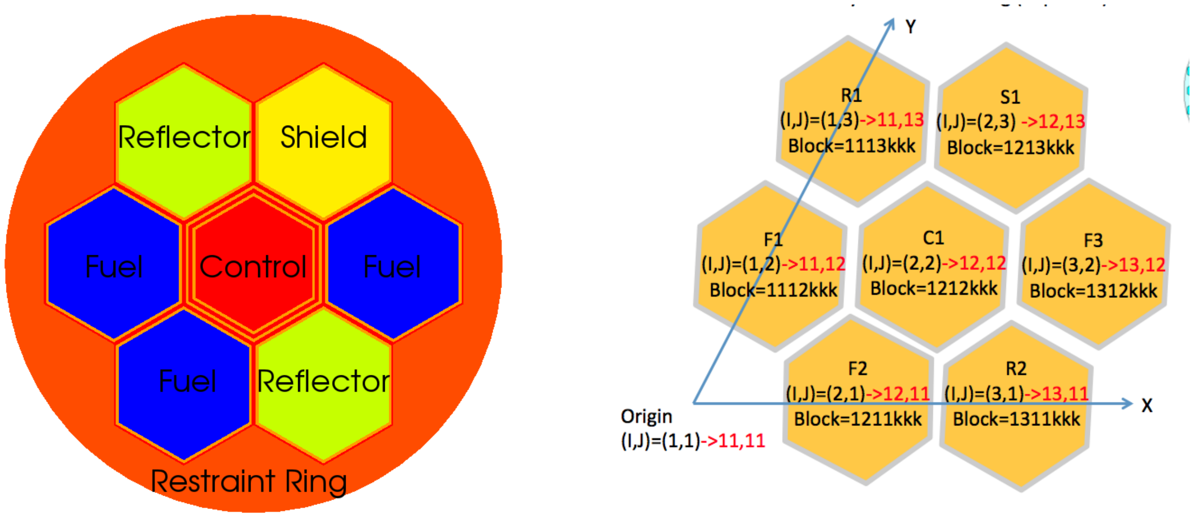
This mini-core consists of 7 assemblies surrounded by a restraint ring. The core contains 3 inner core fuel assemblies, 2 reflector assemblies, 1 shield assembly, and 1 central control assembly (Figures 4.7 and 4.8). This case was intended to test 4 of the assembly types (fuel, reflector, shield, control) present in the ABTR geometry. The CoreGen input file for the 7-assembly core is given in Appendix C.

The small problem size allowed for easier debugging of the mesh via both visualization and manual inspection of input/output. During the specification of this problem, conventions were agreed upon for the mesh block ordering, which was instrumental to streamlining the input generation for the full-core case. All the assemblies used in this model are same as those described for the 199-assembly core (section 4.4.3).

Each of the 4 different types of assemblies has material and boundary condition names prefixed with IIJJ (Figure 4.7b). To overcome the problem of manually creating 3 separate AssyGen files for fuel assemblies, the “CreateMatFiles” keyword was introduced. This keyword creates AssyGen files with name “IJ”.inp and sets the start material and boundary condition numbers based on “IJ.” This is important for the 199-assembly core model, where tens of files of each kind are present. One AssyGen run on the base file that describes all required “IJ”.inp files via CreateMatFiles keyword generates all the input files corresponding to that particular assembly. This enables a numbering scheme that is manageable and helps prescribe temperature, flow rate, and so forth for a particular assembly easily.

Fig. 4.9 highlights the ACLP and TLP regions. The gap between assemblies and restraint ring is kept the same as the full core restraint ring model (see Fig. 4.3 for full-core gap dimensions). In

Fig. 4.9c, load pads on fuel assemblies are highlighted. These load pads are present in all assemblies outside the structural steel covering. The sodium flow region between the load pads of individual assemblies is divided in half and modeled with each assembly separately.



(a)

(b)

Figure 4.7. Mini-core assembly: (a) Configuration; (b) Numbering scheme of assemblies (red).

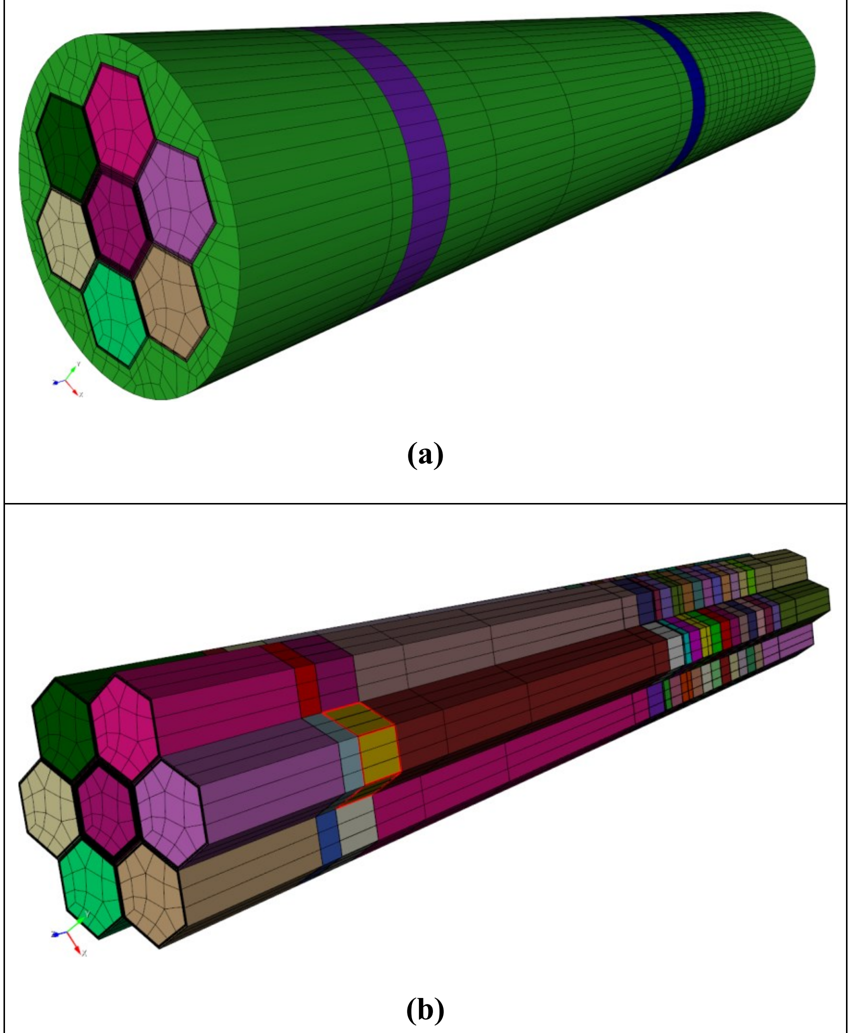


Figure 4.8. Mini-core mesh: (a) elements and blocks; (b) internal assembly blocks used in PROTEUS.

4.4.2 Mini-Core with Fuel Assemblies Only

A second mini-core consisting of 7 fuel assemblies surrounded by a restraint ring was created to provide a more realistic model, while keeping the problem size small (Figure 4.10). The CoreGen input file for creating the mesh for the all-fuel mini-core can be found in Appendix E.

4.4.3 Full Core Mesh

The 199-assembly full core with restraint ring consists of 60 fuel assemblies (24 inner, 30 outer, 6 test), 10 control assemblies, 48 shield assemblies, 78 reflector assemblies, and 3 material test assemblies (which are modeled like reflector assemblies). Fig. 4.11a also shows the detailed configuration with I-J numbering and the number of occurrences of each of the assembly types.

Horizontal lines represent J increasing from top to bottom and slanted vertical lines following the core arrangement represent I from left to right. All 199 assemblies must be modeled independently with different materials in order to enable specification of varying densities, inlet/outlet boundary conditions, and so forth for a particular assembly in the core model. The core is modeled with 4 different assemblies (fuel, reflector, shield, and control).

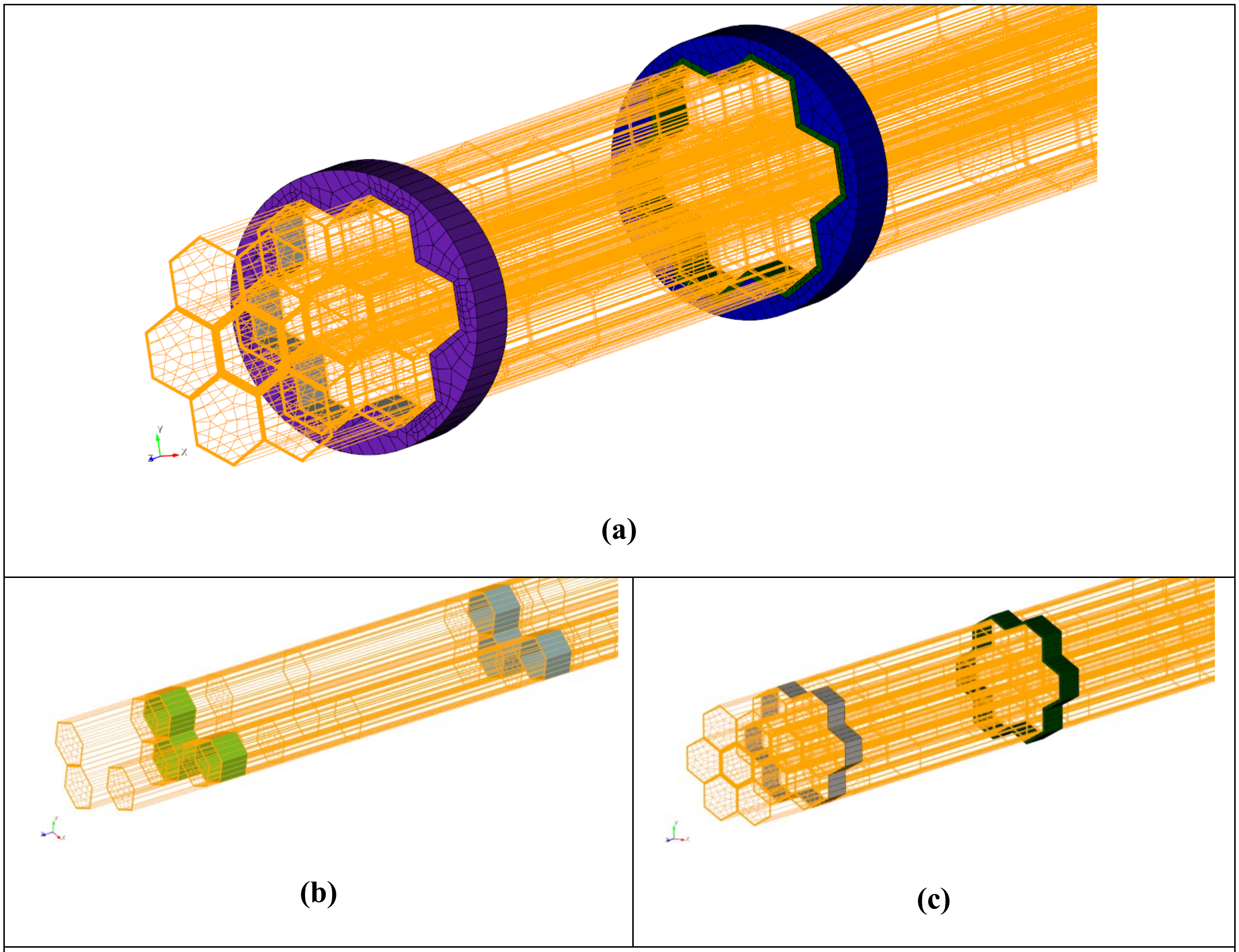


Figure 4.9. Assembly structural components: (a) Restraint rings; (b) Load pads on fuel assemblies; (c) Gap between assemblies and restraint rings.

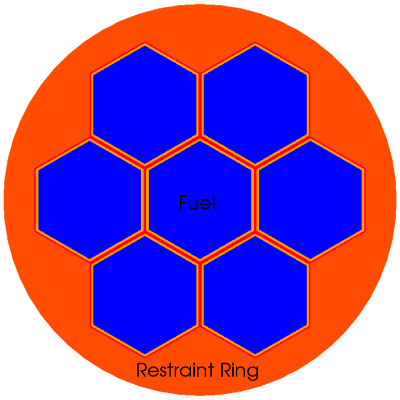
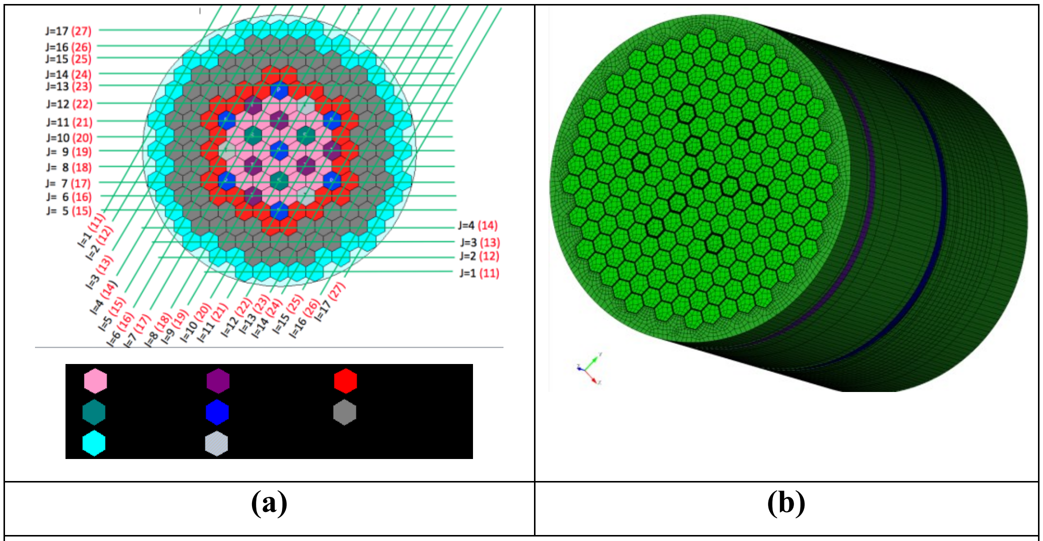


Figure 4.10. Composition map for fuel-only mini-core



(a) **(b)**

Figure 4.11. ABTR full-core configuration: (a) With lines of constant logical I,J assembly regions; (b) With homogenized assemblies, outer covering, and restraint rings at TLP and ACLP locations

The “IJ”.inp files are specified using CreateMatFiles keyword (section 4.4.1). The CoreGen input file used for creating the final mesh (Fig. 4.11b) can be found in Appendix E.

The final homogenized core model is created from assembly meshes (Section 4.3) and the outer covering mesh (Section 4.2). The mesh shown in Fig. 4.11 consists of 800,000 hex elements. The structural mechanics and thermal-hydraulics mesh consists of 1,500 material blocks, whereas the neutronics mesh consists of 7,200 material blocks. The neutronics mesh requires more material blocks along the height of the fuel pins. It takes 30 minutes for CoreGen to create this model on a Linux workstation using 32 processors.

The modeling tools are very flexible. Any assembly in the core can be changed, and various combinations of fuel and other assemblies can be simulated very easily. For example, different outer rings with varying TLP and ACLP gaps can be created by modifying the parameters that generates the restraint rings. The currently available geometry tools can provide a nonhomogenized resolution that taxes readily available computing resources, so simpler, homogenized models were used in the demonstration.

5 Standalone Simulations with Individual Physics Codes

In the SHARP multi-physics simulation framework, the MOAB framework is used to transfer solutions between individual physics codes. During a given time step, each individual code runs a simulation similar to standalone mode, except it uses updated input data given by another physics code. Therefore, a single time step in multi-physics coupling mode is similar to a standalone physics calculation.

Testing the multi-physics input in standalone mode therefore allows the geometry, mesh, and other inputs to be debugged and confirmed with each of the three physics codes: PROTEUS, Nek5000, and Diablo. This section discusses the standalone physics setups, as well as standalone physics results, which illustrate the initial condition for the coupling problem.

Three standalone cases were set up and performed with the individual physics codes in preparation for the full core multi-physics simulation case:

1. A mini-core consisting of 7 assemblies surrounded by a restraint ring. The core contains 3 inner core fuel assemblies, 2 reflector assemblies, 1 shield assembly, and 1 central control assembly. This case was intended to test each of the main ABTR assembly types (fuel, reflector, shield, and control).
2. A mini-core consisting of 7 assemblies surrounded by a restraint ring, where all seven assemblies are fuel assemblies. This case was meant to be a more realistic variation of the first case, while still keeping the problem size small.
3. The third case is the 199-assembly full core ABTR problem with restraint ring, containing 60 fuel assemblies (24 inner, 30 outer, 6 test), 10 control assemblies, 48 shield assemblies, and 81 reflector assemblies (3 of which are surrogates for material test assemblies).

The following sections describe the setup and simulation of these cases using PROTEUS, Nek5000, and Diablo.

5.1 Neutronics (PROTEUS)

The deterministic neutronics code PROTEUS requires the following inputs: mesh, multigroup cross section data, mapping of material data to the mesh (including material assignments, density, temperature, and material models), and algorithm-related criteria (i.e., convergence criteria, angular discretization, etc.). These inputs are illustrated in Figure 5.1.

As described in Section 4, the mini-core and core meshes explicitly (heterogeneously) represent the assembly ducts, load pads, restraint ring, and inter-assembly sodium gap. The assemblies themselves (interior of the duct) were fully homogenized. The geometry was coarsely meshed with linear finite elements in order to keep the number of degrees of freedom reasonably small for the full-core neutronics calculations.

In coupled simulation mode, PROTEUS performs temperature-averaging on a block-wise (not element-wise) basis. The mesh must therefore be defined initially with different blocks in regions where the temperature is likely to change steeply. In this coupling demonstration, the active core regions were axially subdivided into 10 blocks, each of which can have unique temperature and density properties during the coupled simulation. The block-wise temperature averaging therefore increases the number of blocks required in the simulation beyond what is needed for assigning material properties. An algorithm to subdivide blocks internally to the code (rather than a priori in the mesh) is under development.

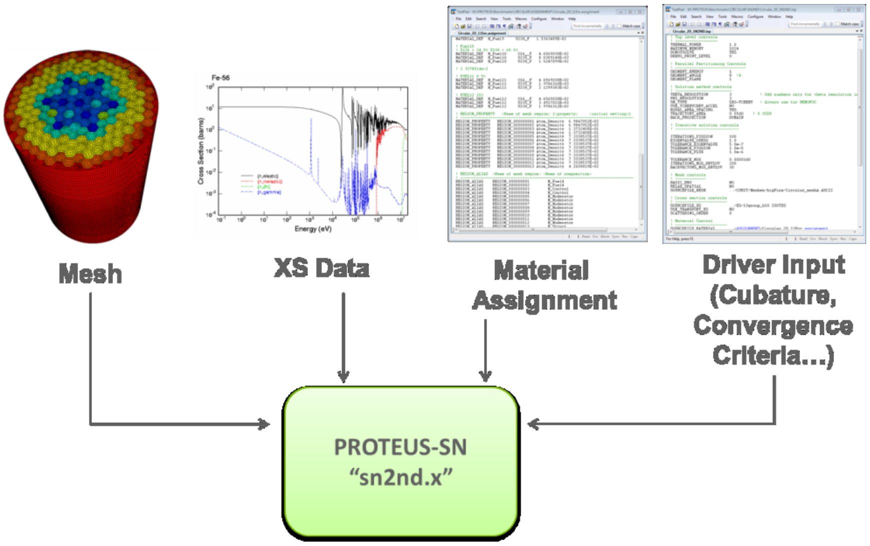


Figure 5.1. Inputs required by PROTEUS.

The MC^2-3 code was used to generate a set of 9-group cross sections used for all the cases. The spectrum collapse procedure was performed individually for each of the homogenized compositions throughout the assemblies (composition is dependent on axial position as well as assembly type) as well as the explicit compositions needed for the cladding, duct walls, load pads, restraint ring, and sodium coolant. The generation of accurate heterogeneous multigroup constants requires significant effort and is outside the scope of this work, so no attempts were made to account for the heterogeneity effects in these problems. Instead, the compositions were analyzed independently in MC^2-3 as infinite homogeneous mediums. The generation of accurate heterogeneous multigroup constants is an important area of ongoing research at Argonne. For this demonstration, the use of approximate infinite medium constants is sufficient. The energy group boundaries of the multigroup cross sections are listed in Table 5.1.

For the material mapping (i.e., assignment) file, homogenized compositions were assigned to the assembly interior regions (inside the duct). Heterogeneous compositions were assigned to the ducts themselves, as well as the inter-duct sodium gap, duct load pads, and restraint ring. Table 5.2 lists the volume fractions for cold and hot conditions calculated based on the known geometry at these conditions provided in the specification document. Note that hot volume fractions were used for the fuel homogenization, and all gas plenums were assumed to be filled with sodium to reduce complications due to time-dependent material changes (level of sodium rising). This should have minimal impact on the neutronics behavior since it is well above the active core. Table 5.3 lists the heterogeneous and homogeneous densities used in the neutronics simulation. Both atom density and mass density are listed.

In coupling mode, PROTEUS queries a property called "MATERIALMODEL" to identify which compositions have sodium and therefore should be influenced by temperature and density feedback. This property is important only in the coupled demonstration and is ignored in standalone mode. Additionally, initial temperatures are assigned to the domain for use in coupled mode. Since Nek5000 provides the actual temperature upon iteration, only rough initial conditions are needed in PROTEUS. The initial temperature was arbitrarily set to 700K (heterogeneous regions) or 705K (homogenized regions).

Table 5.1. 9-Group energy boundaries for multi-group cross sections.

Group	Lower Energy Bound (eV)	Upper Energy Bound (eV)
1	2.2313020E+06	1.4190675E+07
2	8.2085000E+05	2.2313020E+06
3	1.8315634E+05	8.2085000E+05
4	4.0867668E+04	1.8315634E+05
5	9.1188105E+03	4.0867668E+04
6	2.0346827E+03	9.1188105E+03
7	4.5399911E+02	2.0346827E+03
8	5.0434737E+00	4.5399911E+02
9	0.0000000E+00	5.0434737E+00

NOTE: the multigroup data contains P3 scattering cross sections; however, the PROTEUS simulation used only P0 scattering.

Table 5.2. Volume fractions calculated for explicit duct geometry cases.

Homogenized Region	Material	VF (Cold)	VF (Hot)
Homogenized inner, outer, test fuel	Fuel slug	*	*
	HT9 cladding	*	*
	Sodium bond	0.1288	0.0000
	Sodium coolant	0.3188	0.3188
Homogenized lower reflector	HT9 solid pin	*	*
	Sodium coolant area	0.3188	0.3188
Homogenized upper plenum (fuel assembly)	Gas or sodium fill	0.5156	0.6812
	HT9 cladding	*	*
	Sodium coolant	0.3188	0.1532
Homogenized lower and upper structures	Sodium	0.7000	0.7000
	SS-316	0.3000	0.3000
Homogenized control rod follower	Follower	0.4905	0.4895
	Coolant	0.5095	0.5105
Homogenized control rod absorber	HT9 cladding	*	*
	Absorber	0.4032	0.4745
	Sodium bond	0.0712	0.0000
	Sodium coolant	0.3787	0.3833
Homogenized upper plenum	Gas or sodium fill	0.4745	0.4745

Homogenized Region	Material	VF (Cold)	VF (Hot)
(control assembly)	HT9 cladding	*	*
	Sodium coolant area	0.3787	0.3788
Homogenized reflector	Solid HT-9 rods	*	*
	Coolant	0.1187	0.1187
Homogenized shield	HT9 cladding	*	*
	B4C absorber	0.4880	0.6025
	Helium bond	0.1144	0.0000
	Sodium coolant	0.1373	0.1373

**** these values are not open literature***

Table 5.3. Atom and mass densities of heterogeneous and homogenized compositions.

Composition	Atom Density (at/barn-cm)	Mass Density (g/cm^3)
Sodium	2.22720E-02	8.50111E-01
HT9 (duct, load pad)	*	*
SS316 (restraint ring)	7.64344E-02	7.16215E+00
Homogenized lower reflector	6.26935E-02	5.40354E+00
Homogenized inner fuel	3.73370E-02	7.09596E+00
Homogenized outer fuel	3.73346E-02	7.09517E+00
Homogenized test fuel	3.73308E-02	7.11287E+00
Homogenized upper plenum (fuel assembly) – assumed sodium fill	3.20985E-02	1.95705E+00

Homogenized upper structure	3.85210E-02	2.74373E+00
Homogenized control follower	4.88118E-02	3.94302E+00
Homogenized control absorber	6.16426E-02	2.17330E+00
Homogenized upper plenum (control assembly) – assumed sodium fill	3.09829E-02	1.83138E+00
Homogenized shield	7.39764E-02	2.97403E+00
Homogenized reflector	7.45672E-02	6.74109E+00

**** these values are not open literature***

NOTE: the mass density is required in the PROTEUS assignment file for coupled calculations in order to perform density feedback.

The key parameter in the driver input is the angular discretization, which was set to L5T7 after performing angular convergence studies with both the 7-assembly and 199-assembly problems. This corresponds to 48 angles in PROTEUS or 96 physical angles. A mesh refinement study should also be carried out, but it was beyond the scope of this demonstration.

5.1.1 Seven-Assembly Mini-Core (Three Fuel Assemblies)

The first seven-assembly problem is a mini-core comprising seven assemblies taken from the ABTR design. Three fuel assemblies, two reflectors, one shield, and one control assembly are arranged in the configuration (2D view) shown in Fig. 5.2. The three fuel assemblies were assigned the “inner core” fuel composition. Note the duct and inter-duct sodium gap are represented explicitly.

This small test case was formulated to test each of the assembly types (i.e., ensuring that the mesh could be imported and used by each of the physics codes) as well as the restraint ring geometry. It does not represent a realistic reactor core. Seven is the minimum number of assemblies to test all four assembly types in hexagonal geometry. The small problem size is appropriate for a demonstration.

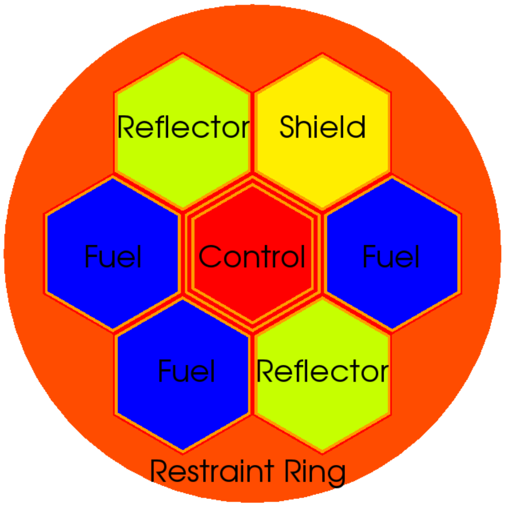


Figure 5.2. Seven assembly mini-core configuration containing three fuel assemblies.

Figure 5.3 depicts the mesh for this case. The thick black lines are actually the outline of the explicitly modeled duct and inter-assembly sodium gap. Void boundary conditions are applied on the cylindrical boundary and on the flat upper and lower surfaces of the core. This mesh consisted of 25,776 elements and 27,625 vertices, which is relatively small for PROTEUS and can easily be run in serial mode.

Legendre-Tchebychev cubature order L5T7 was chosen after performing angular convergence studies on the eigenvalue. The total power was set to 12.5 MWt based on the ABTR power of 250 MWt (about 4.17 MWt per fuel assembly). The eigenvalue converged after 18 iterations to k=0.30562. This small eigenvalue is reasonable because of the large leakage resulting from a small core configuration with no layer of reflector or shield assemblies and only 3 fuel assemblies.

Despite the unrealistic configuration, this problem was very instructive in debugging the mesh and establishing numbering conventions. To demonstrate the data fidelity and visualization capabilities of PROTEUS, a plot of the power distribution in the fuel assemblies is included in Figure5.4. One can observe the qualitatively correct behavior of the cosine-shaped flux distribution in the axial direction, and peaked flux between the two adjacent fuel assemblies.

Figure 5.5 shows the power distribution at the axial slice Z=138.0 cm, which is halfway up along the active core (Z=98.0 to Z=178.0). Figures 5.6 and 5.7 illustrate the fast and thermal flux (groups 1 and 9, respectively) at Z=138.0. The fast flux follows the same pattern as the power distribution – peaked in the fuel assemblies – since fast neutrons are born in these regions. The thermal flux is peaked in the two reflector assemblies as expected.

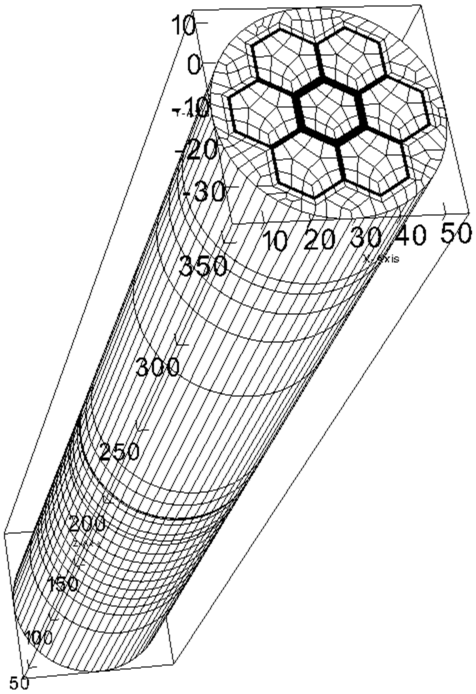


Figure 5.3. Mesh view of seven assembly case with three fuel assemblies.

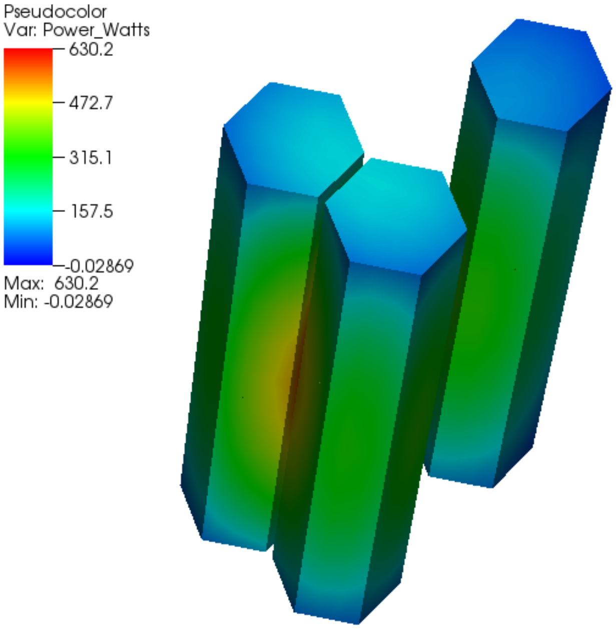


Figure 5.4. 3D Power distribution in the three fuel assemblies for the seven assembly case.

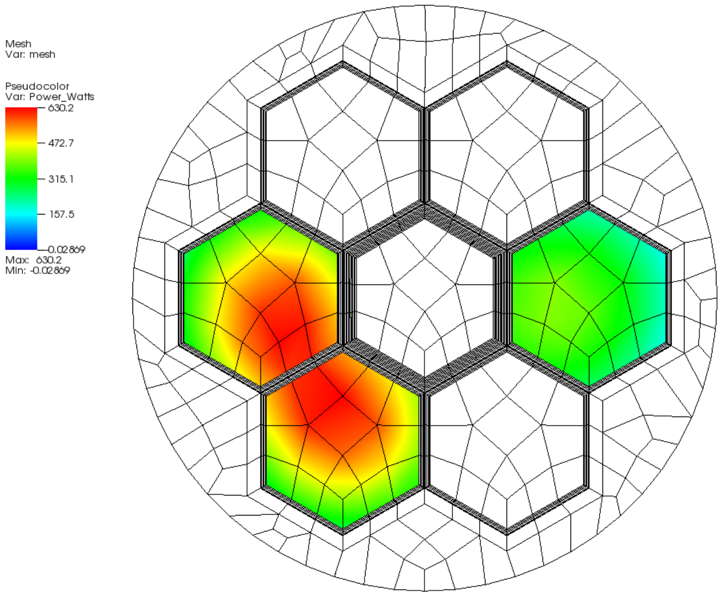


Figure 5.5. Power distribution in fuel assemblies at Z=138.0 (halfway up height of active core).

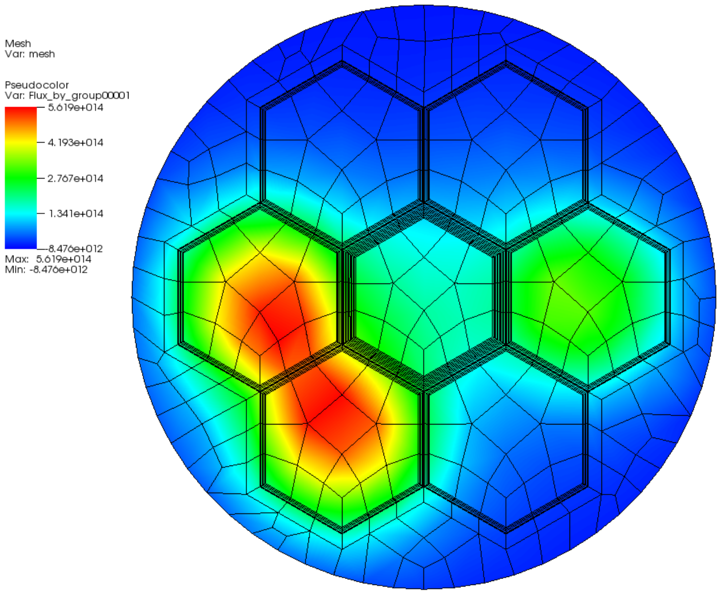


Figure 5.6. Group 1 (fast) flux distribution at Z=138 cm (axial center of active core).

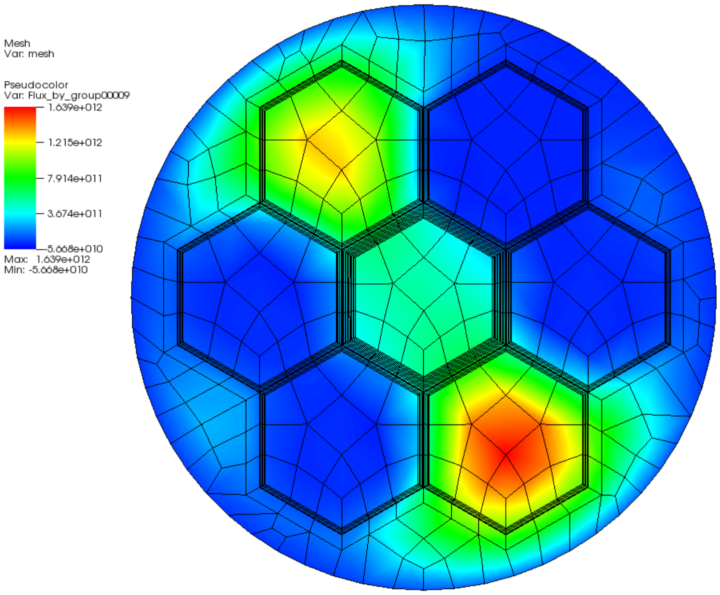


Figure 5.7. Group 9 (thermal) flux distribution at Z=138 cm (axial center of active core).

5.1.2 Fuel-only Mini-core (Seven Fuel Assemblies)

Due to the limited amount of fuel and unrealistically asymmetric power distribution in the previous test case, a second small test problem was defined with seven fuel assemblies (again using inner core fuel). The core map is shown in Fig. 5.8. The mesh for this case had 24,475 vertices and 22,752 elements. The computing time on 2 nodes (8 cores) of the Cosmea cluster at Argonne is about 193 seconds.

The eigenvalue for this case was calculated by PROTEUS to be 0.50939 using L5T7 cubature (same cubature as previous case).

The power distribution in the fuel assemblies is given in Figure5.9 and it follows the expected trend – it is peaked at the axial and radial center point of the core. The subdivision of axial zones needed for temperature interpolation in the active core region is also seen from this figure. Coarse meshing was used (1 element per axial zone). Finer meshing could provide a more detailed solution due to the use of linear finite elements. However, this mesh suffices for the purpose of this demonstration, which is to demonstrate solution transfer and coupling. Figure 5.10 illustrates the power distribution at Z=138.0 in the active core.

The 3D flux distributions for group 1 (fast) and group 9 (thermal) are plotted in Figures 5.10 and 5.11. They follow the expected patterns, i.e., fast flux primarily in the fuel and thermal flux primarily in the upper plenum. There is very little thermalized flux in this case due to the absence of reflectors and shields. The fuel assemblies are bordered only by the restraint ring.

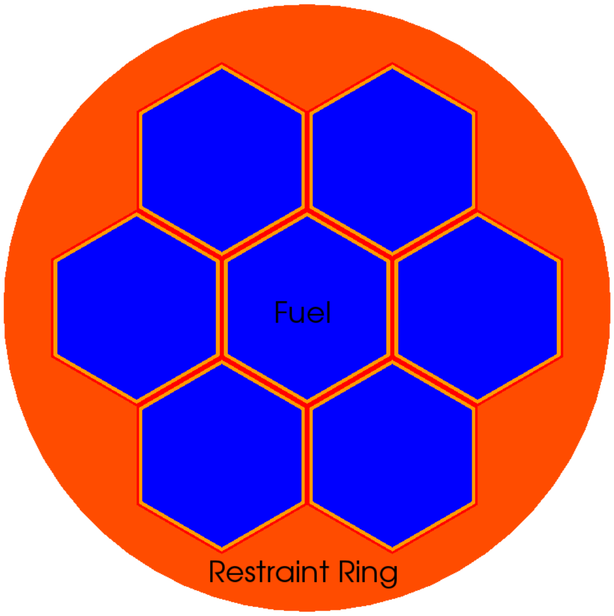


Figure 5.8. Composition map for 7-assembly core with 7 fuel assemblies.

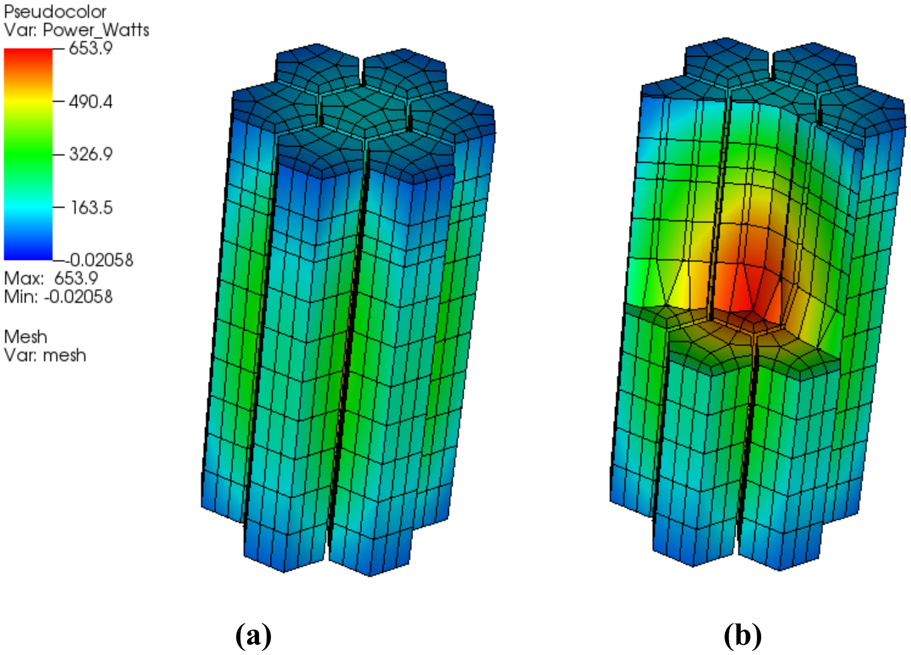


Figure 5.9. (a) Power distribution in active core region of fuel assembly; (b) Cutaway at Z=138.0 cm (axial midplane of fuel) to show interior detail

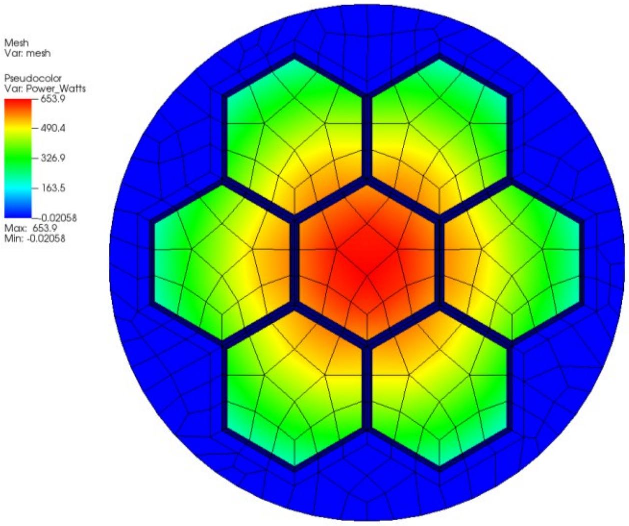
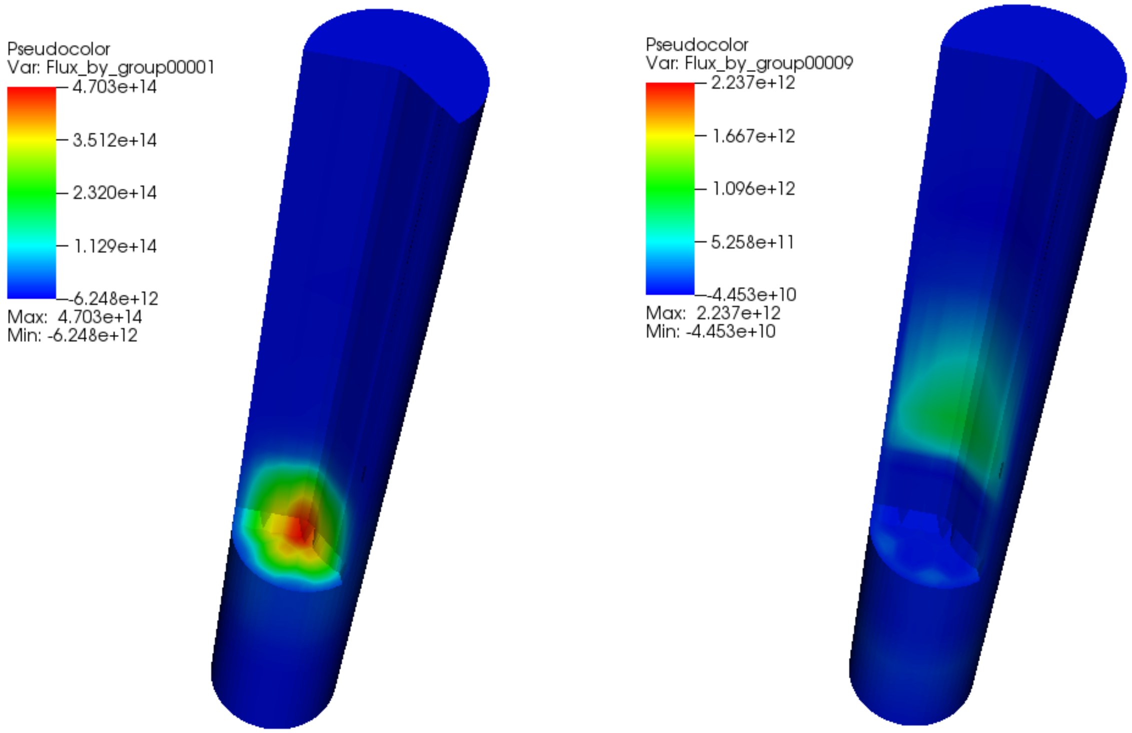


Figure 5.10. Power distribution for the seven fuel assembly case at Z=138.0 (axial midplane of fuel).



(a) **(b)**

Figure 5.11. Cutaway of the fuel-only mini-core showing (a) Group 1 flux and (b) Group 9 flux.

5.1.3 Full Core Problem

The full core ABTR problem has 199 assemblies in total, including 60 fuel assemblies. The total power is set to 250 MWt based on the specification. The fuel assemblies comprise three different types: inner core, outer core, and fuel test assemblies, which differ only by fuel composition. The core map (Figure 5.12) illustrates an explicit representation of the double duct in the control assemblies.

The full core mesh has 825,125 vertices and 789,696 elements. Combined with 48 modeled angles in PROTEUS and 9 energy groups, this problem consists of 356.5 million space-angle-energy degrees of freedom. Parallelization is necessary in order to reduce memory per processor requirements as well as the computer wall-clock time. PROTEUS is highly parallelizable, and 64 processors were used to run the full-core problem. The total wall-clock time using 64 processors was about 27 minutes for the standalone calculation.

The results of an angular convergence study are presented in Table 5.4. The calculation showed little sensitivity in the eigenvalue once a cubature of order L5T7 or higher was used. The L5T7 eigenvalue was 1.00269. Note that the control rods are withdrawn above the active core in this case.

Figure depict the power distribution in the fuel assemblies of the full-core ABTR model. Figure5.14 shows the 3D distribution in the entire active core region, and Figure5.15 shows the same data sectioned along the core centerline to expose interior features. The power distribution is peaked at the radial and axial center of the core, and the z-dependence appears to be approximately cosine shaped. Figure5.16 highlights the radial power pattern at Z=138.0 cm (axial midplane of the fuel).

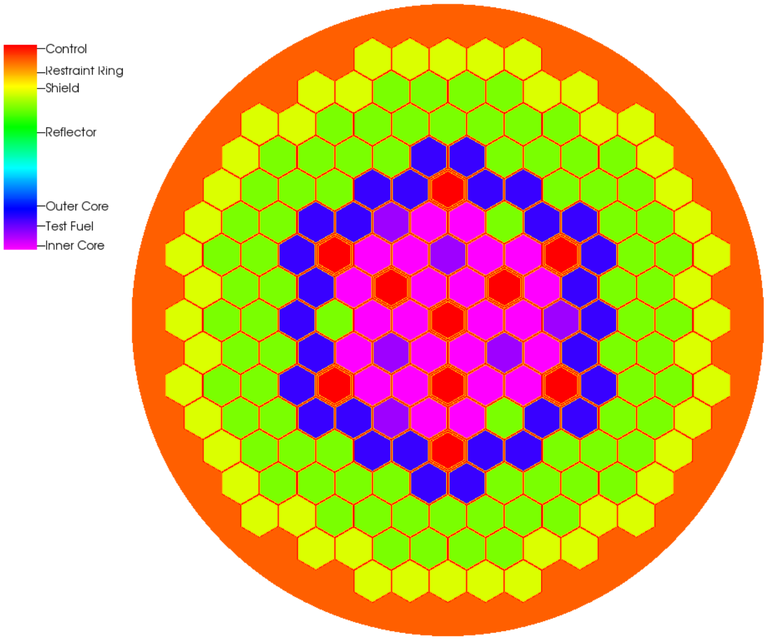


Figure 5.12. Full-core composition map showing explicit ducts.

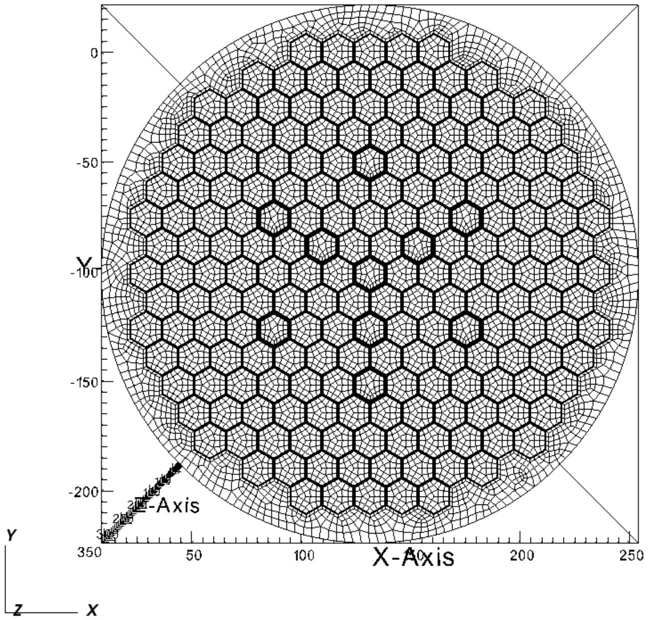


Figure 5.13. Top view of the full-core mesh.

Table 5.4. Eigenvalue Angular Convergence of Full-Core Case.

Cubature Order	**No. of Angles (=2*(L+1)*(T+1))**	**No. of Angles (PROTEUS) (=(L+1)*(T+1))**	**K-effective**
L1T1	8	4	1.00104
L3T5	48	24	1.00249
L3T7	56	28	1.00239
L3T9	80	40	1.00236
L5T7	96	48	1.00269
L7T7	128	64	1.00277
L7T9	160	80	1.00274

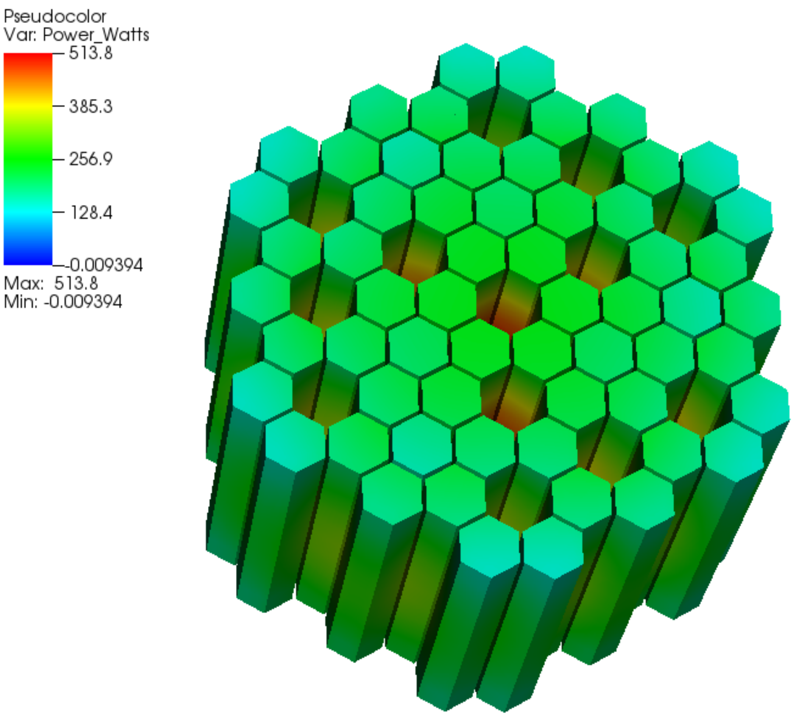


Figure 5.14. Power distribution in fuel assemblies along entire active core height.

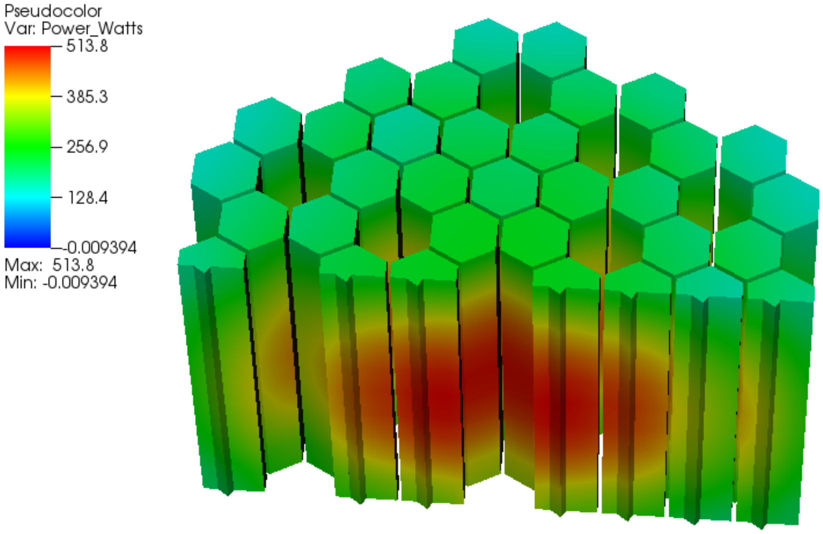


Figure 5.15. Centerline cross section of power distribution in fuel assemblies.

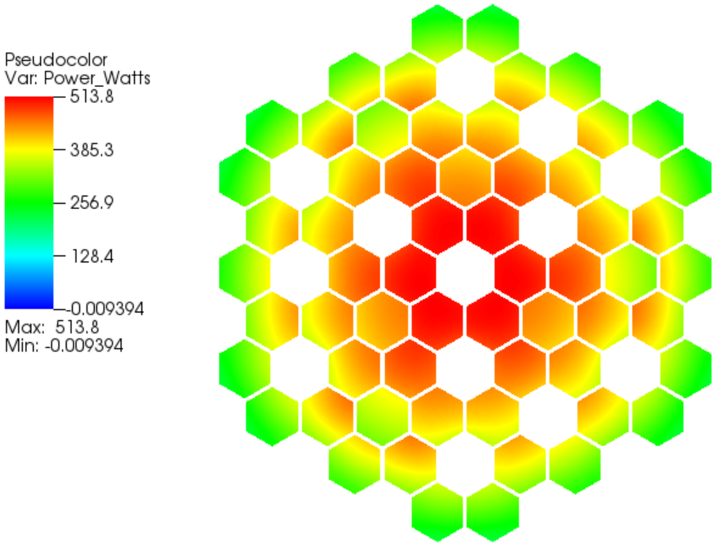


Figure 5.16. Full-core fuel assembly power distribution at Z=138.0 cm (axial midplane of fuel).

Figures 5.17–5.20 illustrate the group 1 (fast) and group 9 (thermal) flux dependence in the full-core problem. Figure5.17 shows the group 1 flux at Z=138.0 cm and along the core centerline. The fast flux in the fuel assemblies is particularly evident. Note the fast flux immediately drops off by orders of magnitude above and below the active fuel height, as well as in the reflector and shield assemblies. This is expected due to the increased moderation outside the active core. Figure5.18 shows the group 9 flux along the same geometrical cross section of the core. Note the thermal flux peaking in the first layer of assemblies around the active core, i.e., the reflector assemblies. The reflector assemblies clearly serve their purpose of scattering neutrons back towards the core, slowing them down during the process. Additionally, strongly peaked thermal flux is seen just beneath the active core due to the lower reflector located axially beneath the fuel regions. This material similarly reflects neutrons back upwards towards the fuel.

Figures 5.19 and 5.20 illustrate the group 1 and group 9 fluxes along the plane Z=138.0 projected to 2D for easier visualization. Again, the fast flux is peaked in the fuel assemblies as expected, and the thermal flux peaks in the ring of reflector assemblies surrounding the active core.

In summary, these standalone cases validated the correctness of the model and confirmed the fundamental physics at play, particularly for the full-core case.

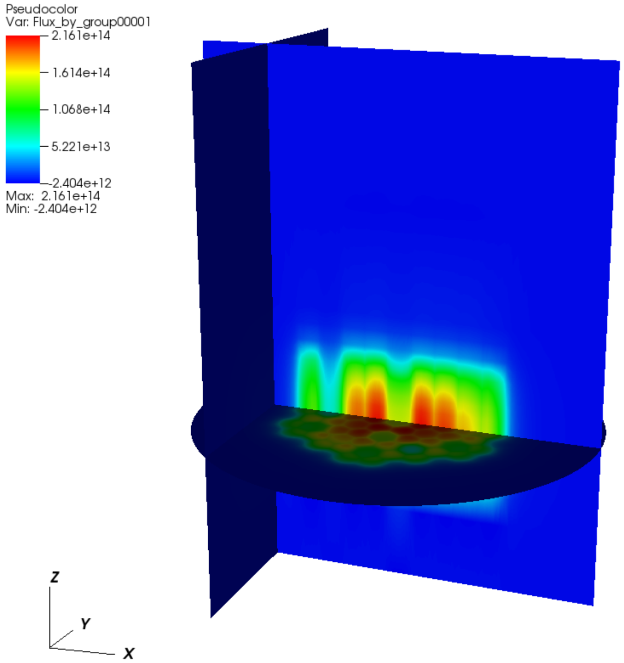


Figure 5.17. Group 1 flux distribution in the full core, cut away at Z=138.0 cm and along the core centerline.

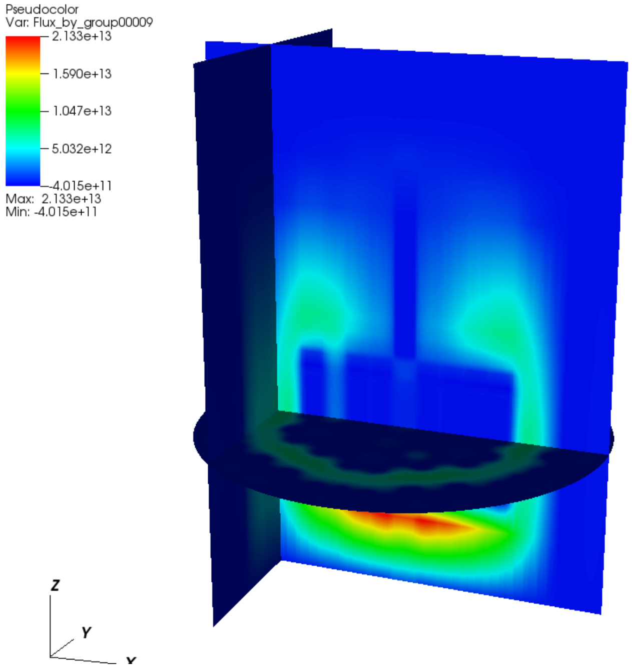


Figure 5.18. Group 9 flux distribution in full core, cut away at Z=138.0 cm and along the core centerline.

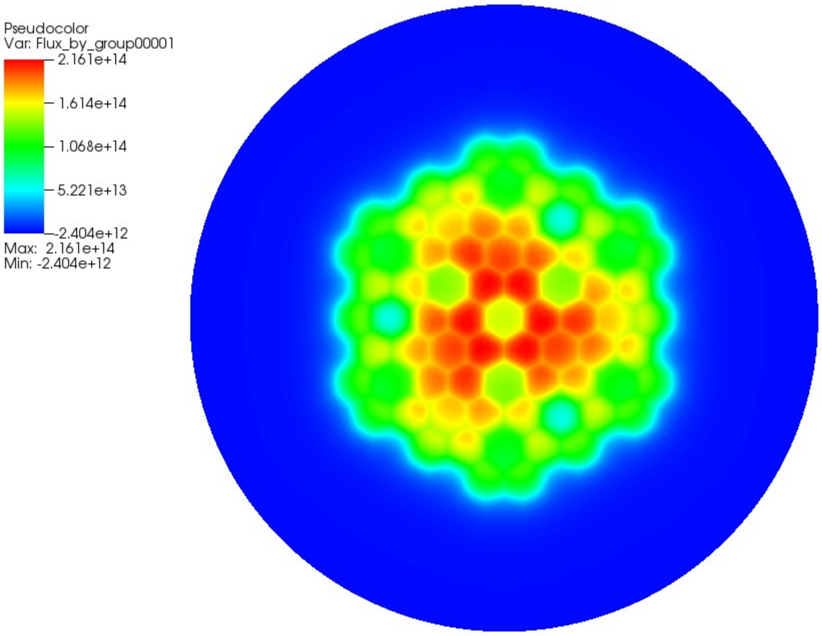


Figure 5.19. Flux in energy group 1 at Z=138.0 cm (axial midplane of fuel).

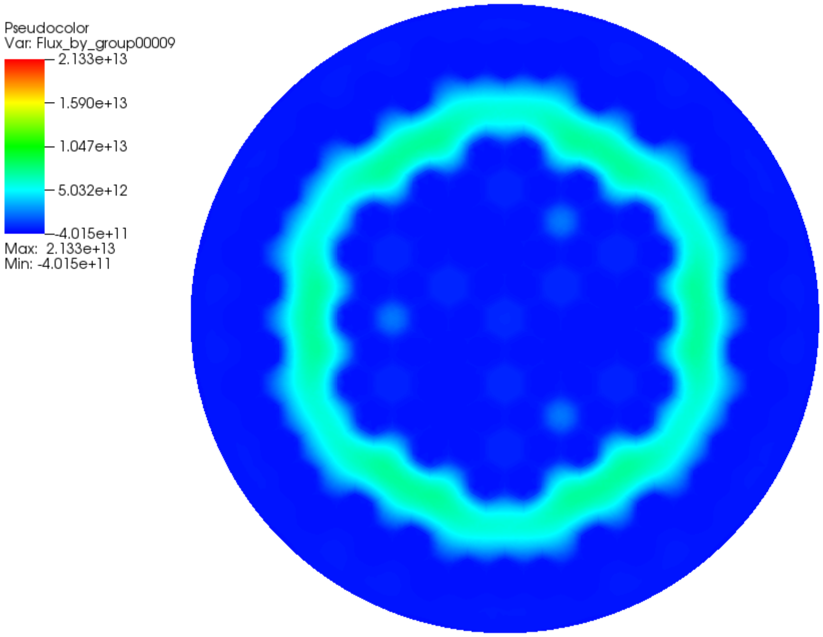


Figure 5.20. Flux in energy group 9 at Z=138.0 cm (axial midplane of fuel).

5.2 Thermal Hydraulics (Nek5000)

The multi-physics spectral element code Nek5000 has been coupled to MOAB as a part of the SHARP thermal-hydraulic module.

Standalone (uncoupled) simulations were performed for 2 cases, the all-fuel 7-assembly mini-core and the full core of 199 assemblies. The all-fuel mini-core model has 29,472 elements and a polynomial order of N=2. Figure 5.21 shows the spectral element mesh with $(N+1)^3$ grid points in each element. The mesh has 14 fluid and 36 solid blocks, which were reduced in Nek5000 to 3 types of material: liquid sodium, structure, and solid sodium. The solid sodium, which is a solid material with the same thermal conductivity as the coolant, is used to fill in the interstitial spaces in bypass channels between adjacent assemblies. If the thin elements in these interstitial spaces were modeled as a fluid, the time-step size would need to be reduced considerably due to the Courant-Friedrichs-Lewy constraint. This simplification does not affect the final results significantly because flow rate through the bypass assemblies is low.

Figure 5.22 shows vertical velocity distribution in the steady-state case for the fuel-only mini-core. The heat generation region in this uncoupled run was modeled as uniform heat source in an axial location that served as a surrogate for input from PROTEUS. Also, the mini-core modeling led to development of a new Nek5000 numbering and boundary condition procedure. The temperature distribution is shown in Figure 5.30(a).

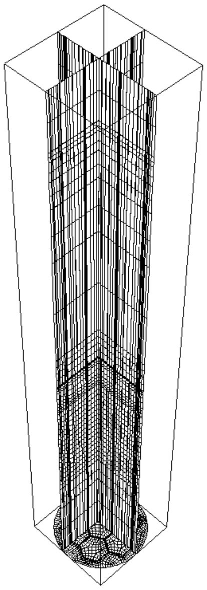


Figure 5.21. Nek5000 mesh for the fuel-only (7-assembly mini-core).

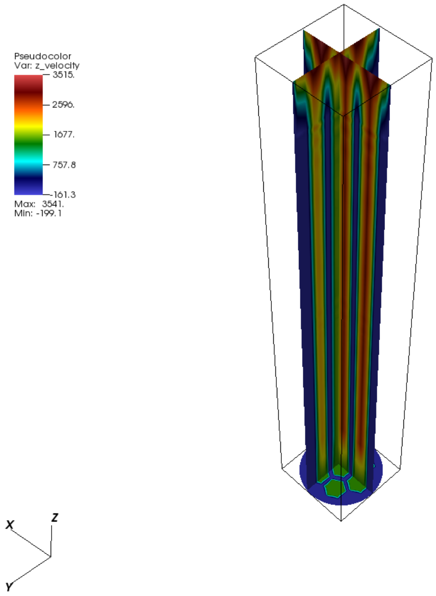


Figure 5.22. Nek5000 steady-state vertical velocity for the uncoupled fuel-only (7-assembly mini-core).

Preparation for the coupled simulation included development of a set of routines that interrogate the Nek5000 mesh and locate internal boundaries between solid and fluid elements in order to automatically prescribe no-slip boundary conditions for the velocity field. These routines significantly reduced the time needed for case setup – from weeks to days for the full core of 199 assemblies.

Another novel development was a modification of Nek5000-MOAB interface to allow large numbers for boundary-condition Neumann sets. The original implementation of Nek5000-MOAB coupling required rigid adherence to the spaces in the Nek5000 parameter file (.rea) and restricted the Neumann set numbering to up to 5 digits. This worked well for the fuel-only mini-core but failed when the numbering scheme mimicked the full-core case that had 7-digit Neumann sets. The Nek5000-MOAB interface now accepts 7-digit sets and is backwards compatible.

To further facilitate the setup procedure for the Nek5000 parameter file (.rea), a generic Mathematica script was developed to write a list of assemblies into a set of files with Cubit block identifiers and corresponding Nek5000 identifiers for fluid and solid elements separately. The Nek5000 identifier for each fluid part of the assembly is unique, while the identifiers for the rest of the assemblies are numbered uniformly. This permits the full, easy, and efficient control of boundary condition prescription for the assembly inlets and facilitates future modeling of various transient conditions in the full core, such as a loss of coolant due to a blockage.

Figure 5.23 shows the mesh for the full-core uncoupled simulation, and Figure 5.24 illustrates the initial transient field of vertical velocity. To reduce the computational cost, the vertical extent of the domain was reduced by use of a special outflow boundary condition (“nozzle”) that prevents reversed flow at the outflow boundary. The outflow nozzle was validated using the 2010 OECD/NEA T-junction benchmark.

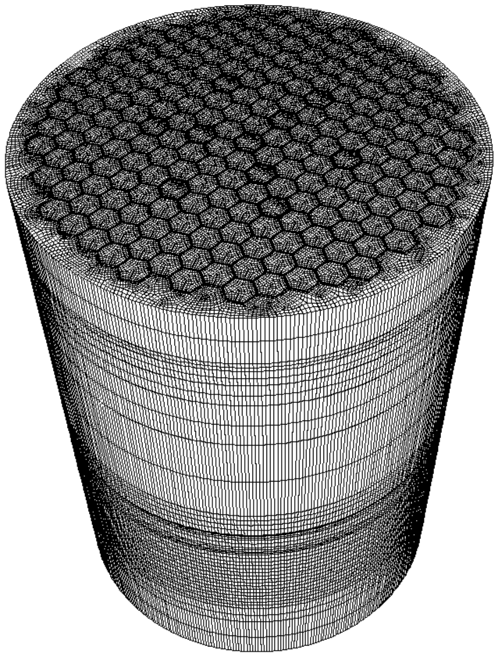


Figure 5.23. Nek5000 mesh for the full-core case.

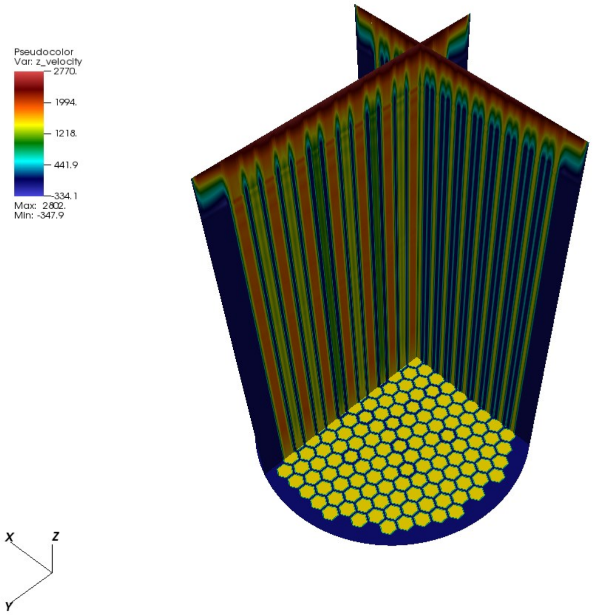


Figure 5.24. Vertical velocity in initial transient Nek5000 uncoupled calculations for full-core case.

5.3 Structural Mechanics (Diablo)

The multi-physics code Diablo is capable of modeling solid mechanics, heat transfer, advection/diffusion, and electromagnetics but in this demonstration is used only for solid mechanics calculations, e.g., temperature data provided by Nek5000 is used to predict the resulting deformations. Diablo requires the following inputs: (1) the mesh file providing spatial definitions, communicated as either an EXODUS or MOAB file; (2) an assembly file containing global data such as material and friction model information, time step, solution algorithm-related controls (convergence criteria, linear solver, nonlinear iteration technique), and input/output-related controls (restart, plotting, coupling to external codes/data); and (3) a "subassembly" file that maps specific spatial quantities in the mesh description (nodesets, sidesets, element blocks) to Diablo element sets incorporating material model assignment, interface contact sets, and boundary conditions. The "subassembly" file also contains controls for individual element and contact sets, such as integration method, penalty stiffness scaling, etc. The collective problem definition is illustrated in Fig. 5.28.

As described in Section 4, mini-core and core meshes explicitly (heterogeneously) represent the assembly ducts, load pads, restraint ring, and inter-assembly sodium gap. The assemblies themselves (interior of the duct) are fully homogenized. The geometry is coarsely meshed with linear finite elements to keep the number of degrees of freedom reasonably small for the full-core case.

Diablo requires only a subset of the overall domain, the Structural Components:

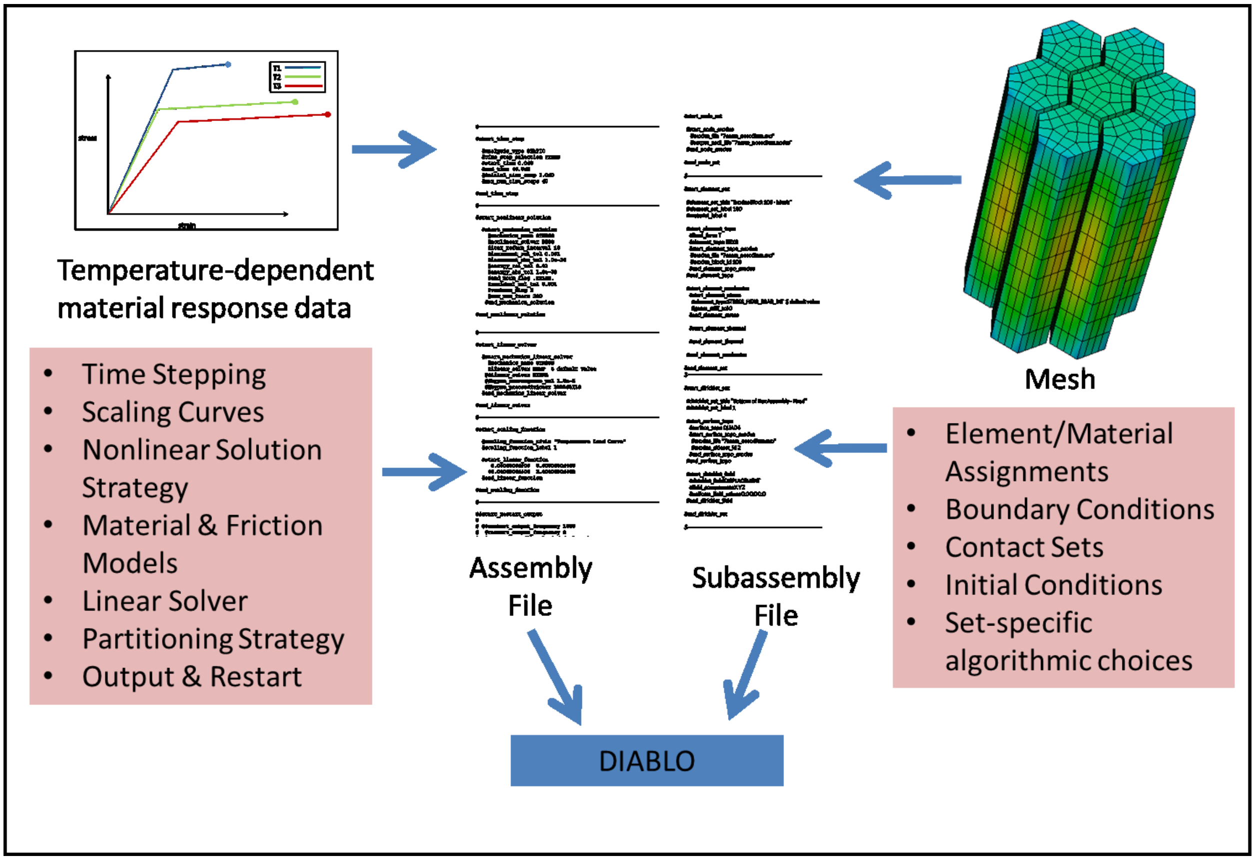
- Duct wall
- Homogenized duct interior
- Load pads
- Restraint rings

In contrast, Nek5000 and PROTEUS consider a different subset of the domain, the HydroNeutronic components:

- Duct wall
- Homogenized duct interior
- Upper sodium
- Outer sodium (including load pads and restraint rings modeled as sodium fill)

Only the duct wall and homogenized duct interior are common to both the Structural Mechanics and the HydroNeutronics definitions.

Diablo uses the temperatures provided by Nek5000 to calculate the deformation of the Structural Components. The deformations calculated by DIABLO for the common core components are then processed with MOAB's mesh smoothing capability to construct deformed meshes for the HydroNeutronic components for both Nek5000 and PROTEUS. Note that the common components do not include the load pads or restraint rings as modeled by Diablo, since it is possible that they will contact each other during the simulation, and the resulting "zero-thickness" elements in the gap between them would not be admissible in a Nek5000 or PROTEUS simulation.



Temperature-dependent material response data

- Time Stepping
- Scaling Curves
- Nonlinear Solution Strategy
- Material & Friction Models
- Linear Solver
- Partitioning Strategy
- Output & Restart

Assembly File

Subassembly File

Mesh

- Element/Material Assignments
- Boundary Conditions
- Contact Sets
- Initial Conditions
- Set-specific algorithmic choices

DIABLO

Figure 5.25. Problem definition required by Diablo.

During ongoing development of the MOAB smoothing machinery, Diablo is being run to directly compute all required deformations for the HydroNeutronic components. This requires using duplicated components in select, limited regions. Specifically, the load pads and restraint rings are modeled as two different materials simultaneously. This allows Diablo to compute a valid deformed mesh for the HydroNeutronic components while at the same time correctly taking into account mechanical contact between load pads and between load pads and the restraint rings.

Hence, a composite mesh is used to compute deformations and a smooth mesh across the entire computational domain:

- Duct wall
- Homogenized duct interior
- Load pads, with mechanical contact
- Restraint rings, with mechanical contact
- Upper sodium

- External sodium
- Load pads (duplicated), modeled as external sodium
- Restraint rings (duplicated), modeled as external sodium

Only a subset of the composite mesh is communicated to Nek5000 and PROTEUS, consisting of:

- Duct wall
- Homogenized duct interior
- Upper sodium
- External sodium
- Load pads (duplicated), modeled as external sodium
- Restraint rings (duplicated), modeled as external sodium

At this stage, very simple structural material models are being used. The duct walls, restraint rings, and load pads are all modeled as finitely deforming elastic stainless steel. Diablo Material Model 4 (finite deformation elastic/plastic with temperature-dependent properties) is being utilized, but in the purely elastic, temperature-independent mode. The homogenized duct interior, external sodium, and upper sodium are modeled as very soft linear elastic material. See Table 5.5 for the material properties used in the Diablo simulations.

Table 5.5. Material properties for Diablo simulations.

Material	**Density** **(g/cm^3)**	**Young's Modulus** **($Dyne/cm^2$)**	**Poisson's Ratio**	**Thermal Expansion Coefficient**
Duct wall (SS)	1.746E+04	1.93E+10	0.29	1.227E-5
Load pad (SS)	1.746E+04	1.93E+10	0.29	1.227E-5
Restraint ring (SS)	1.746E+04	1.93E+10	0.29	1.227E-5
Homogenized interior	1.746E+04	1.93E+4	0.29	1.227E-5
External and upper sodium	1.746E+04	1.93E+4	0.29	1.227E-5

Boundary conditions, initial conditions, and contact definitions are required to complete the analysis. In the actual structure, a complicated socket/nozzle arrangement exists at the base of each assembly. This allows for a limited amount of hex can rotation to occur at the base before contact occurs with the socket, constraining subsequent rotation. The details of this joint are highly dependent on manufacturing tolerances, can and socket aging/swell, and refueling insertion force. For this demonstration, the following boundary condition was used:

Duct walls and duct homogenized interior fixed (ux=uy=uz=0) at Z=Zmin

The composite mesh needed no boundary condition for the restraint rings – they float up and down vertically as forced by the thermal expansion of the outer sodium, which keeps them lined up in the axial direction with the load pads. They are strong enough in the radial direction to withstand expansion forces caused by contact between the rings and the duct walls. Future work may include direct modeling of the surrounding core barrel to which the restraint rings are attached.

The chosen initial temperature condition (300K) is representative of the actual deformation of the structure. An initial temperature closer to the average operational temperature (a value of 550K has been suggested) would result in a smaller perturbation to the initial mesh. Ultimately, if convergence of the multi-physics coupled solution becomes a problem, under-relaxation techniques may be applied to the deformation (the output of the solid mechanics simulation) to allow the coupled system to smoothly approach convergence.

A mortar contact algorithm is used where each load pad, as a “slave,” is restrained from penetrating the surrounding load pads and the restraint rings. Two contact sets per duct assembly are therefore required (14 contact sets for the 7-assembly model and 398 for the 199-assembly model). Currently, a penalty/augmented Lagrange algorithm with a tolerance of 1.0E-2 on the total interface contact force is being used, and frictionless contact is assumed. Future work will include investigation of the influence of friction, but as the interface is filled with liquid sodium, the effect of friction is probably small.

Standalone structural deformation calculations are possible, using temperatures either from Nek5000/PROTEUS bi-physics simulation or through the construction of synthetic temperature fields. For the 7-assembly models, preliminary temperature distributions were available from coupled Nek5000/PROTEUS runs with the undeformed geometry, and these were used for the standalone demonstration. For the full-core case, Nek5000/PROTEUS runs were not yet available and the deformation was driven by a synthetic temperature field.

5.3.1 Seven-Assembly Mini-Core (Three Fuel Assemblies)

The first seven-assembly problem is a mini-core comprised of seven assemblies taken from the ABTR design. Three fuel assemblies, two reflectors, one shield, and one control assembly are arranged in the configuration (2D view) shown in Figure 5.26. The three fuel assemblies were assigned the “inner core” fuel composition. The duct and inter-duct sodium gap are represented explicitly.

This small test case was formulated to test each of the assembly types (i.e., ensuring that the mesh could be imported and used by each of the physics codes) as well as the restraint ring geometry. It does not represent a realistic reactor core. Seven is the minimum number of assemblies to test all four assembly types in hexagonal geometry. The small problem size is appropriate for a demonstration.

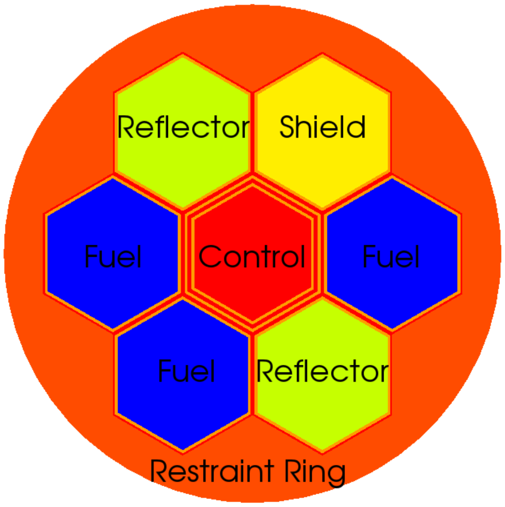


Figure 5.26. Seven-assembly mini-core configuration with three fuel assemblies.

Figure 5.27 depicts the mini-core mesh for this case. The thick black lines are actually the outline of the duct wall and inter-assembly sodium gap. This mesh consisted of 25,776 elements and 27,625 vertices, which is relatively small for Diablo and can easily be run in serial mode. When running with the composite geometry (with the load pads and constraint rings duplicated along with certain of their nodes), there are 26,264 elements and 28,685 vertices. Figure 5.28 shows the model stripped of the outer sodium (material model 8) and the duplicated elements/nodes. Regions defined as material model 1 represent the duct wall. Material 2 is the region of the upper (TLP) restraint ring, here modeled as stainless steel. Material 3 is the region of the upper (TLP) load pads, also modeled as stainless steel. Material 4 is the lower (ACLP) restraint ring, also modeled as stainless steel. Material 5 is the region of the lower (ACLP) load pads, here modeled as outer sodium. Material 6 is the homogenized duct interior (not seen in the figure), and material 7 is the upper sodium. Figure 5.29 shows the duplicated element blocks, which consist essentially of the structural restraint system: Material 9 is the upper (TLP) load pad, material 10 is the lower (ACLP) load pad, material 11 is the lower (ACLP) restraint ring, and material 12 is the upper (TLP) restraint ring, all duplicated and modeled as external sodium fill.

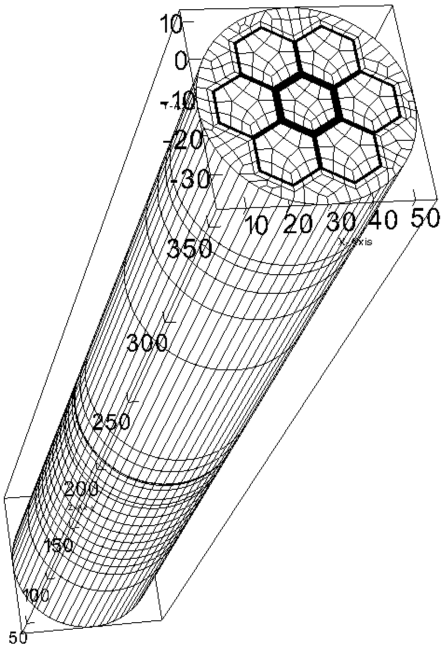


Figure 5.27. Mesh view of seven-assembly mini-core with three fuel assemblies.

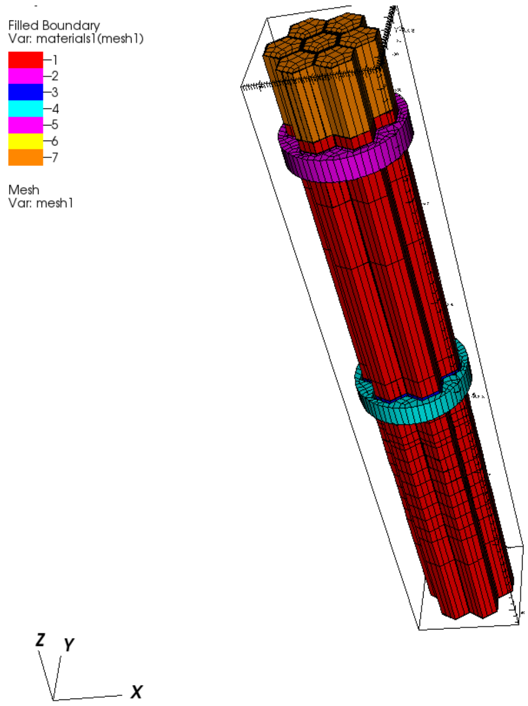


Figure 5.28. 3D View of restraint rings, hex cans, and upper sodium with external sodium removed.

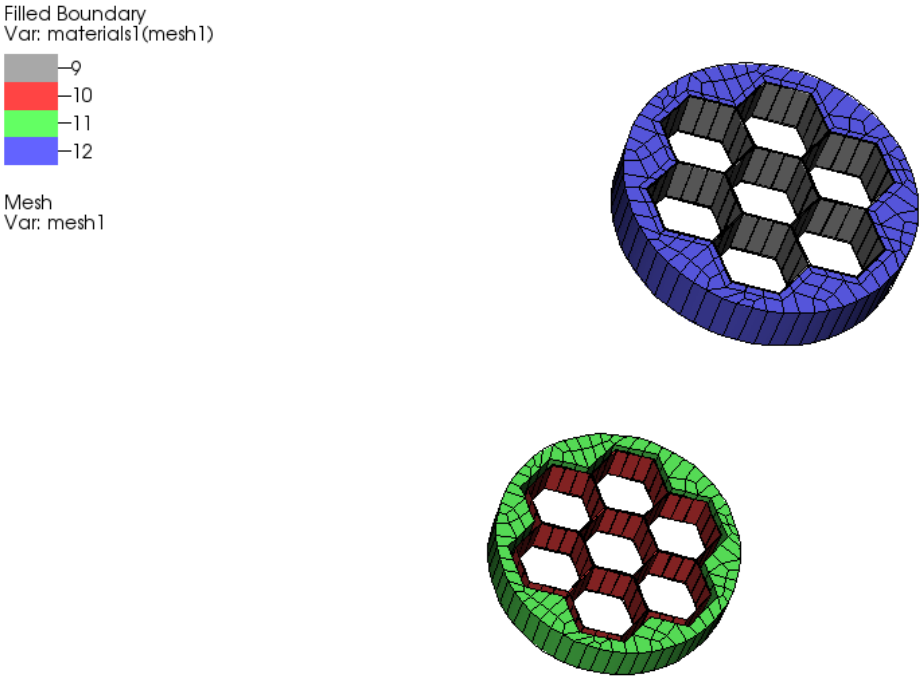


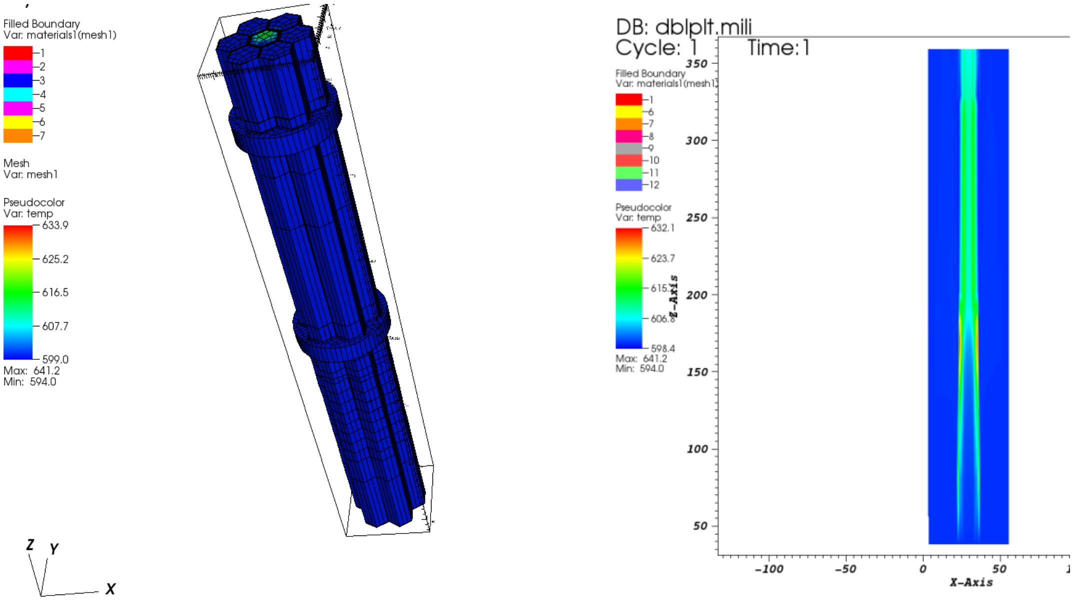
Figure 5.29. Restraint rings and load pads.

Figure 5.30 shows the temperature distribution loaded from the initial Nek5000/PROTEUS run. Note that most of the elevated temperature is in the central control assembly. The maximum temperature over the entire mesh is 641K and the minimum is 594K, a range of only a little over 40K. The initial temperature for the Diablo run was set to 300K. Figure 5.31 illustrates the deformed geometry on both the full mesh and a wireframe representation of only the structural components laid over the undeformed mesh. Figure 5.32 shows the X and Z displacements for a slice through the full mesh through Y-symmetry plane.

5.3.2 Fuel-only Mini-core (Seven Fuel Assemblies)

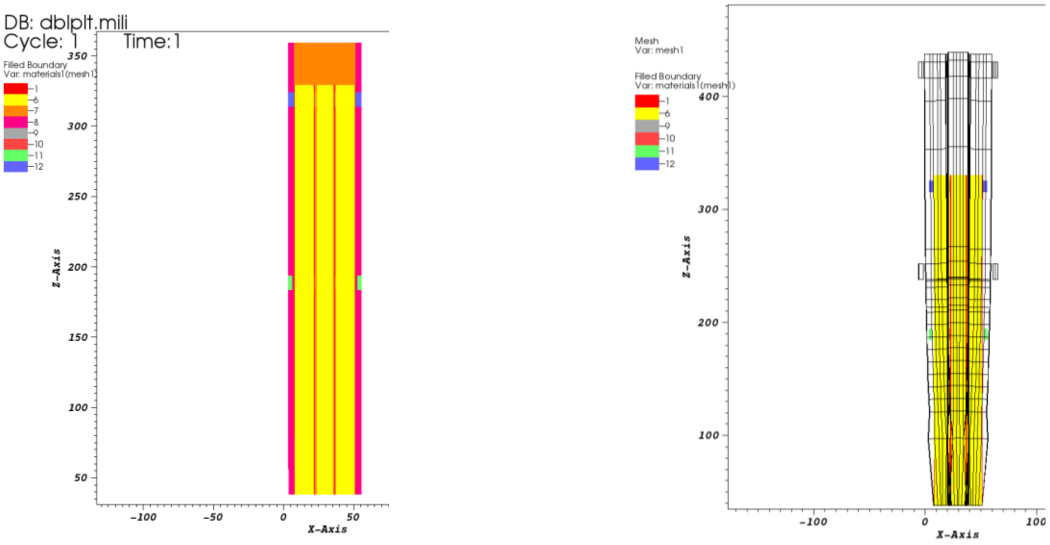
Due to the limited amount of fuel and unrealistically asymmetric power distribution in the previous mini-core case, a second small test problem was defined with seven fuel assemblies (again using inner core fuel). The core map is shown in Figure 5.33, and the resulting temperature profile used as input to Diablo is shown in Figure 5.34.

The displacements for this mesh were also computed and a sample exaggerated set of displacements is shown in Figure 5.35. Note that significant bowing occurs near the upper restraint ring. The upper sodium deformation is largely driven by the fact that it is extremely soft, and the temperature has dropped off at the very top compared to the central core region. Figure 5.36 shows the X- and Z-displacements for a slice through the Y-symmetry plane, for the full mesh.



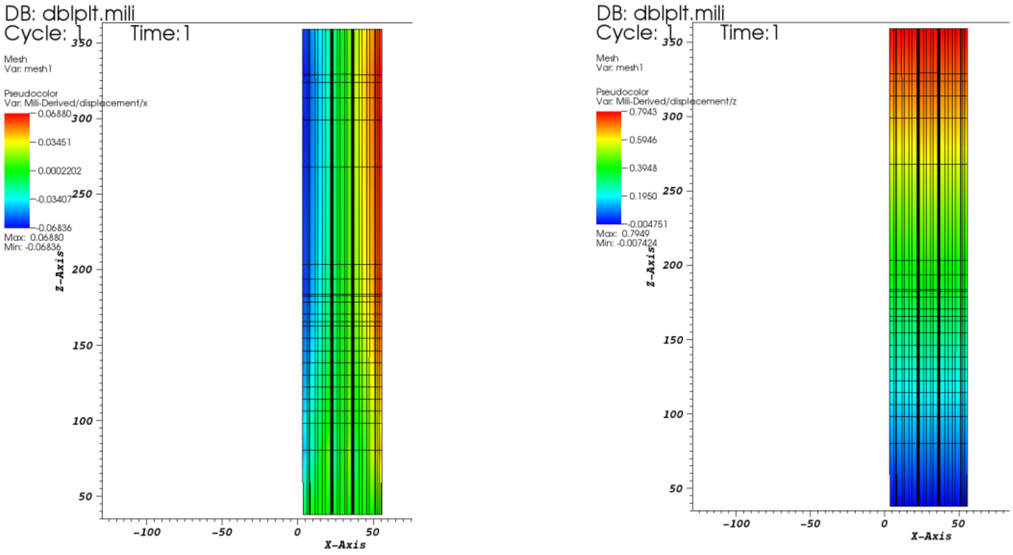
(a) **(b)**

Figure 5.30. Temperature distribution from initial Nek5000/PROTEUS run for the 7-assembly mini-core: (a) Structural components; (b) Slice through the Y-symmetry plane (with outer sodium).



(a) **(b)**

Figure 5.31. Deformed geometry through the Y-symmetry plane of the 7-assembly mini-core: (a) Full mesh; (b) Structural components only (as a wireframe) superimposed over an undeformed mesh.



(a) **(b)**

Figure 5.32. X and Z displacements through the Y-symmetry plane of the 7-assembly mini-core: (a) X displacements; (b) Z displacements

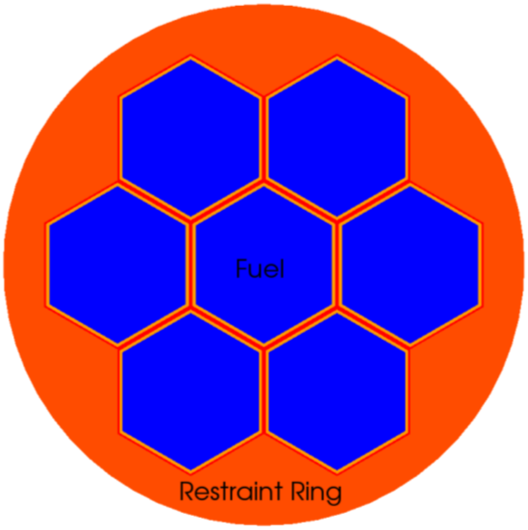
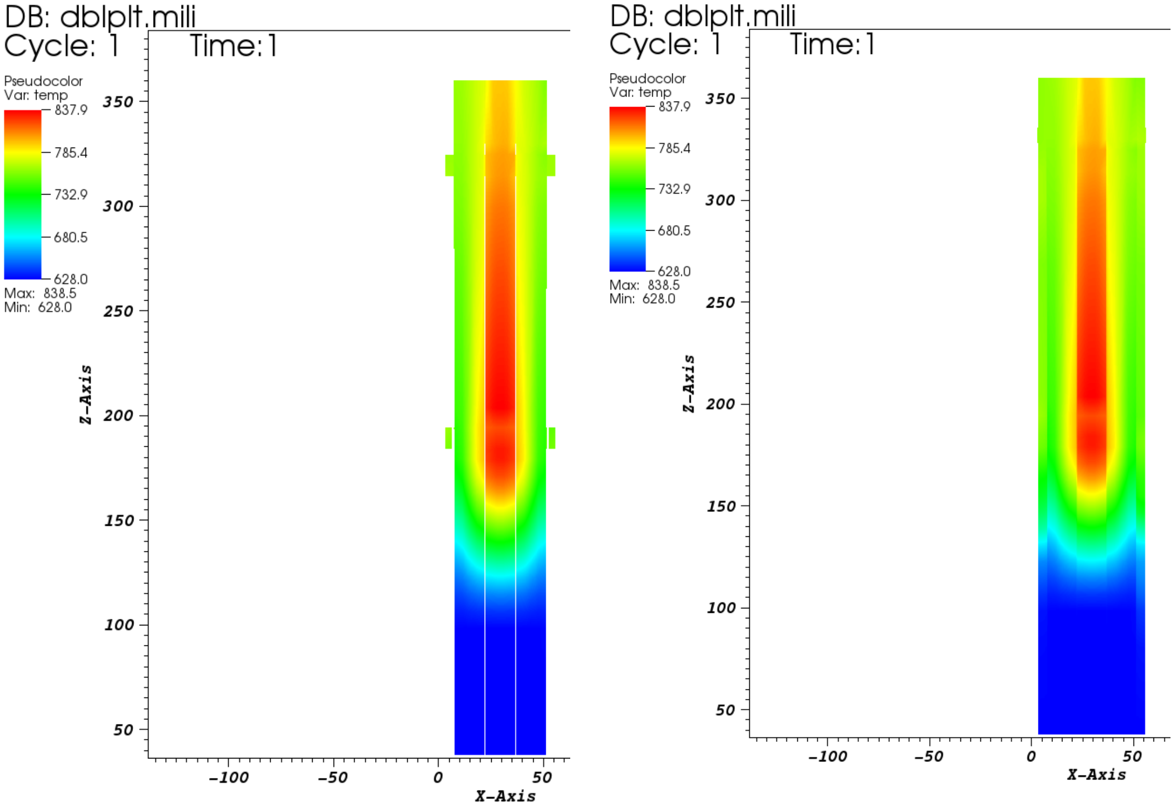
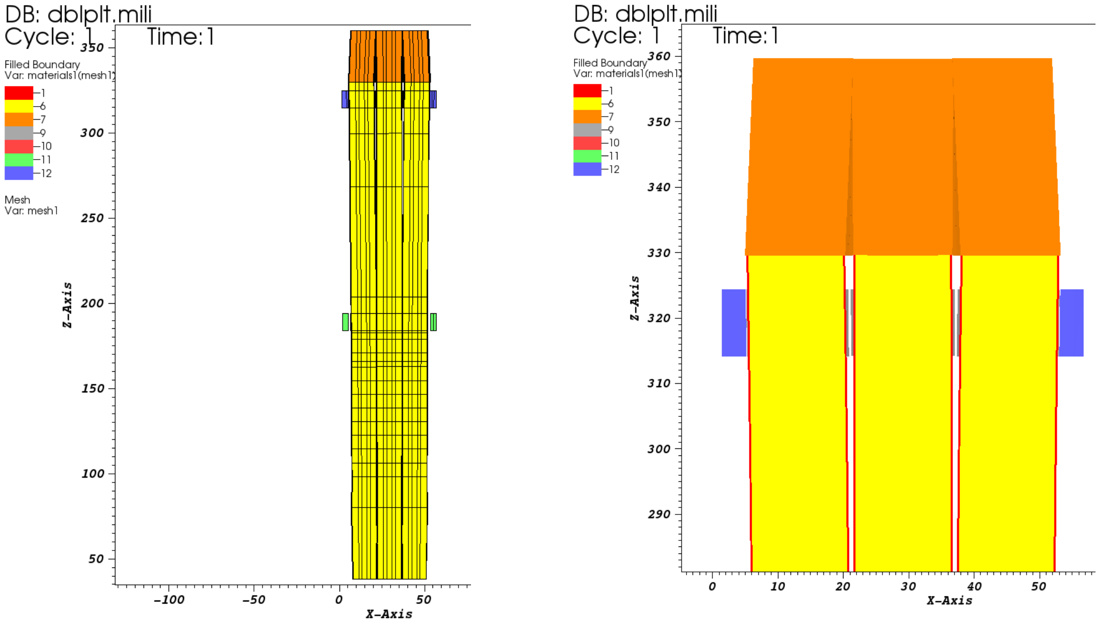


Figure 5.33. Composition map for fuel-only mini-core with 7 fuel assemblies.



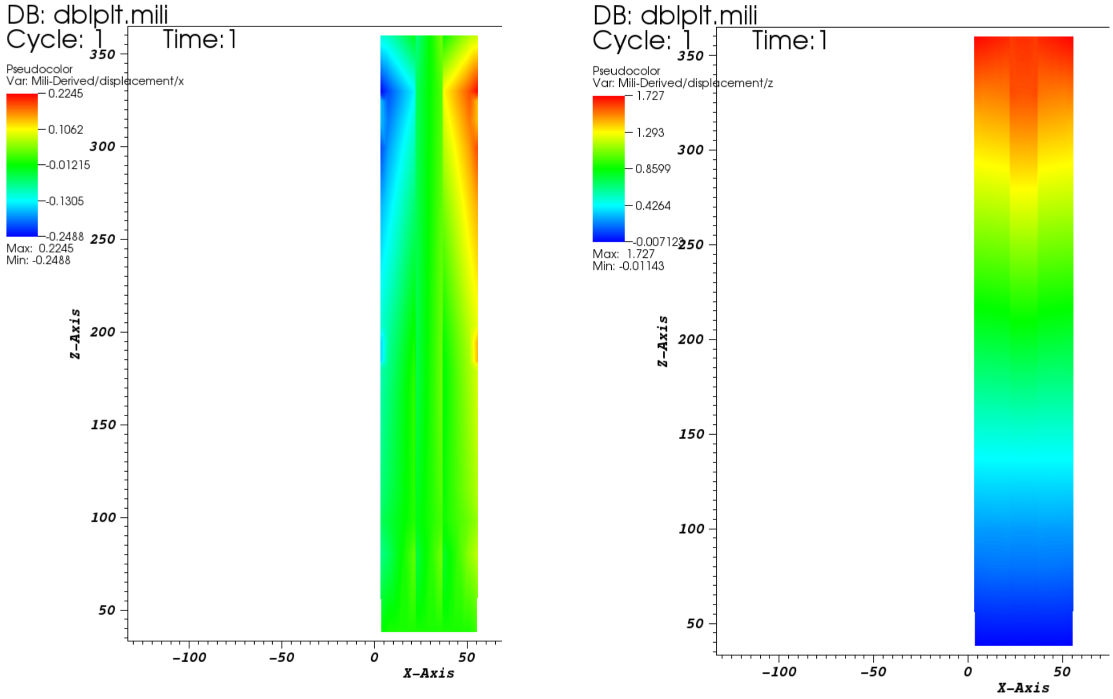
(a) **(b)**

Figure 5.34. Temperature distribution in the fuel-only mini-core, sliced through the Y-axis: (a) Structural components only; (b) Full mesh.



(a) **(b)**

Figure 5.35. Displaced (bowed) mesh through the Y-symmetry plane of the fuel-only mini-core (displacements exaggerated 10×): (a) Structural components; (b) Enlargement of the TLP



(a) **(b)**

Figure 5.36. X and Z displacements through the Y-symmetry plane of the fuel-only mini-core: (a) X displacements; (b) Z displacements.

5.3.3 Full-Core Simulation

The full-core case has 199 assemblies in total, including 60 fuel assemblies. The total power is set to 250 MWt based on the specification. The fuel assemblies are comprised of three different types: inner core, outer core, and fuel test assemblies, which differ only by fuel composition. The core map (Figure 5.37) explicitly represents the double duct in the control assemblies.

The full-core mesh has 825,125 vertices and 789,696 elements. As in the 7-assembly mini-core, duplicate nodes/elements result in a slightly larger problem size when running the composite model. Parallelization is necessary to reduce memory per processor requirements as well as the computer wall-clock time. Diablo is highly parallelizable, and 32 processors were used to run the full-core problem. The total wall-clock time using 64 processors was less than 15 minutes for the standalone calculation. When contact interfaces become active, the problem becomes much more computationally intensive, and it is expected that 64-128 processors will be necessary to keep the simulation time down to less than an hour.

Figures 5.38–5.40 show the mesh with material assignments from three vantage points. The synthetic temperature profile (discussed at the beginning of section 5.3) is illustrated in Figure 5.41. Figure 5.42 shows the deformed configuration, exaggerated by a factor of 100, with the temperature profile laid on top.

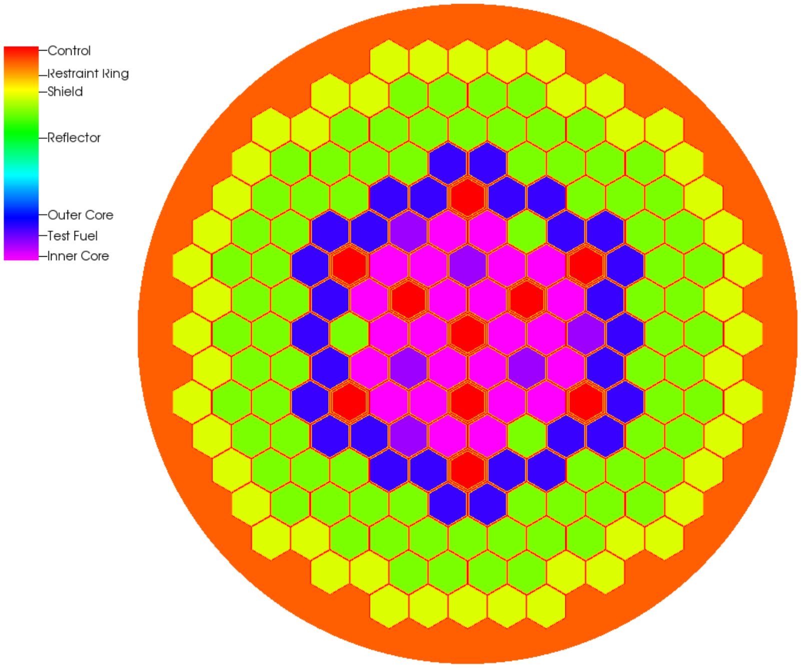


Figure 5.37. Full-core composition map showing explicit ducts.

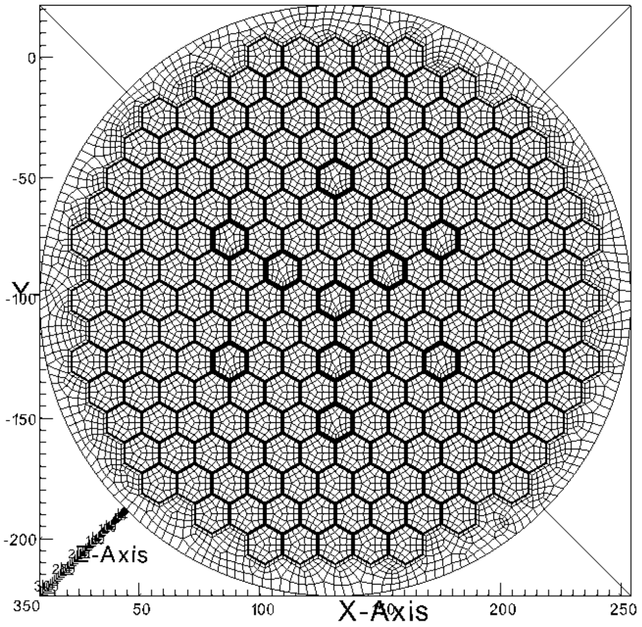


Figure 5.38. Top view of full-core mesh

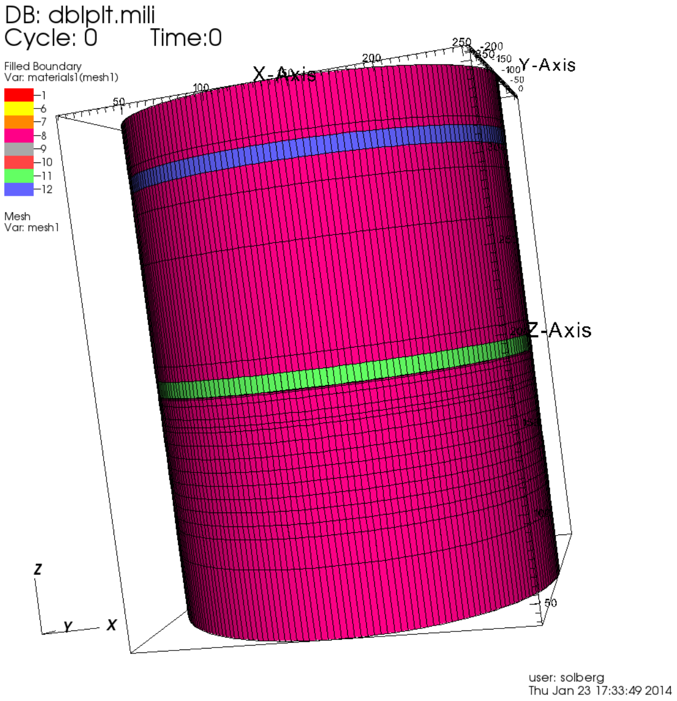


Figure 5.39. Full-core mesh side view.

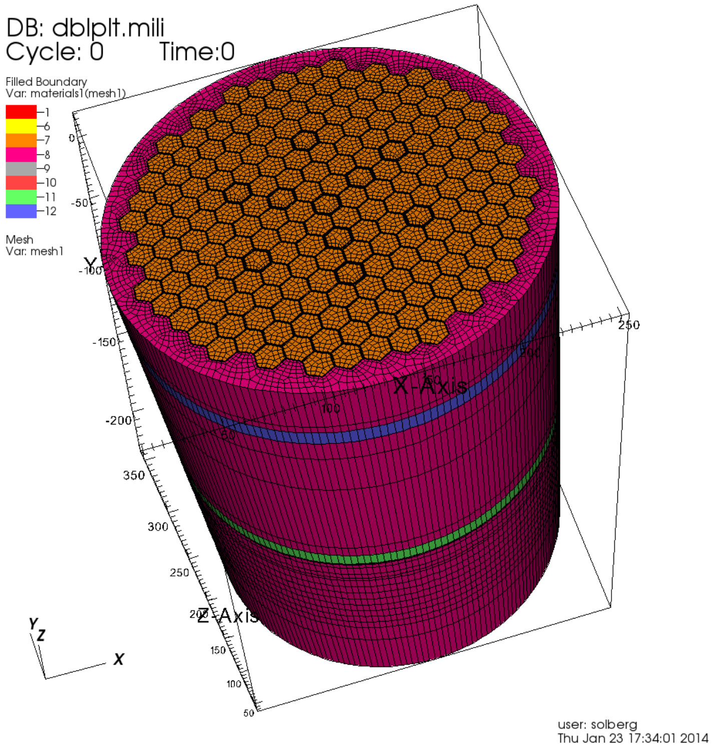


Figure 5.40. Full-core mesh, oblique view.

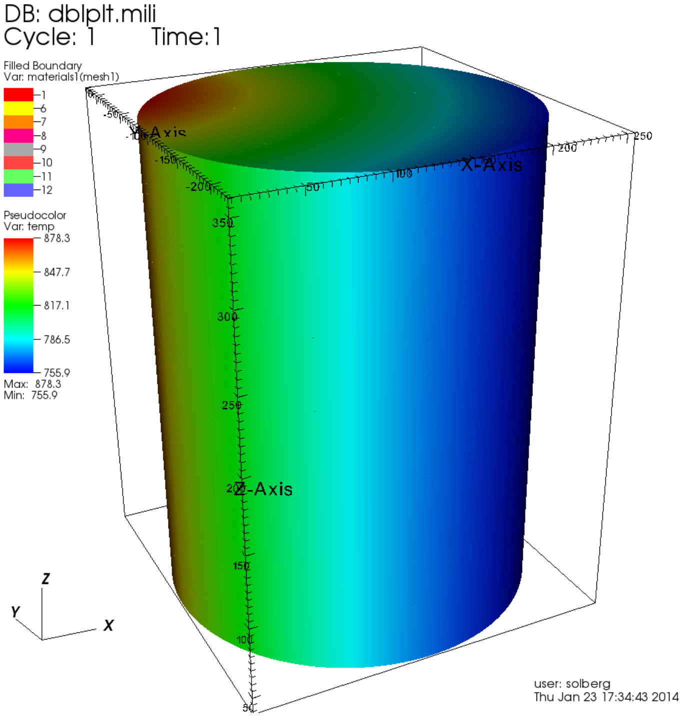


Figure 5.41. Full-core synthetic temperature distribution.

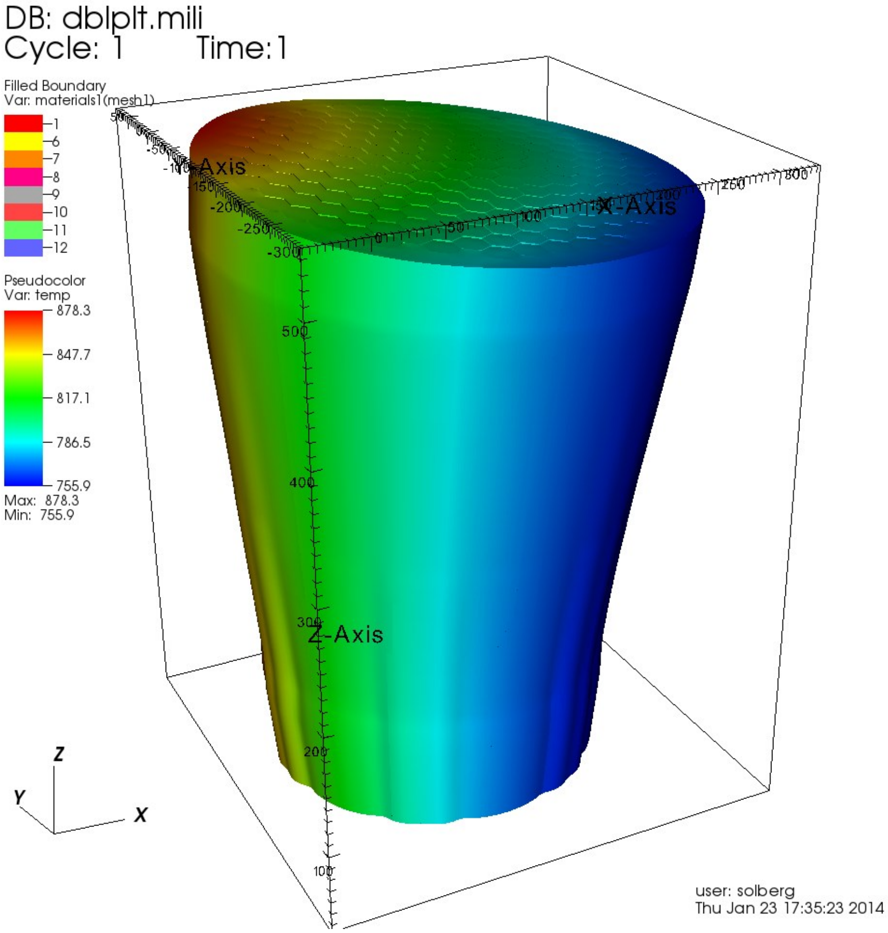


Figure 5.42. Full-core displacements due to synthetic temperature distribution (exaggerated 100×).

6 Coupled Multi-Physics Simulations

A fully coupled simulation of the three physics models was conducted for the 7-assembly mini-cores in preparation of a full-core coupled simulation. An intermediate step was necessary to demonstrate the feasibility of the coupling and for debugging purposes. It is important to note that, as stated in section 1, this work represents the first instance of incorporating structural mechanics feedback in a SHARP coupled simulation. In fact, to our knowledge, this effort represents the first successful three-dimensional coupled simulation including neutronics, thermal-hydraulics, and structural mechanics for an SFR.

6.1 General Approach

The coupling procedure follows what is outlined in section 3. In each SHARP iteration, CouPE drives PROTEUS and Nek5000 to perform a coupled thermal-hydraulics/neutronics simulation. After convergence is achieved, the deformation due to thermal expansion is computed using Diablo and the mesh is deformed for all three models. A new Nek5000/PROTEUS iteration is then performed after inputs are updated. Details of each individual physical model are given in section 5.

A series of calculations with different fuel composition and different models for the density feedback was performed. Cases are listed in Table 6.1. All three cases are models of the 7-assembly mini-core model described in section 5. Case A is the model with four assembly types (see Figure 2.5a), while cases B and C are for the fuel-only version (see Figure 2.5b). The coupled simulations for case A are not particularly interesting. The fuel loading is highly asymmetric (which is inconsistent with any known SFR design), yielding correspondingly skewed power profiles and unrealistic changes in core geometry. Therefore, this section focuses on Cases B and C. The distinction between cases B and C is in the update of the density and material properties in the PROTEUS neutronics module.

Table 6.1 List of coupled simulation cases for the 7-assembly mini-core simulations.

Case	Composition	Material Density Update in PROTEUS	No. of Iterations
A	4 types (3 fuel assemblies)	No	3
B	Fuel only	No	5
C	Fuel only	Yes	4

Structural mechanical deformation of the mesh leads to changes in element (and block) volumes. If the original material densities (g/cm^3) are applied to the modified PROTEUS mesh, conservation of mass would be violated. This leads to an error in computing the macroscopic cross sections. PROTEUS has the capability to update number densities based on volume changes when coupled in SHARP. However, this feature was not used in these simulations because Diablo is not yet fully integrated into SHARP. Therefore, the material densities and compositions must be updated manually between iterations based on the volume changes in the mesh. This manual update of the material densities was performed only for case C; some error is introduced in cases A and B due to the inconsistent material mass. This is indicated in the third column of Table 6.1.

A python script was written to parse and update the original assignment file based on the volume changes resulting from the deformed mesh information. This script automatically updates certain parts of the original PROTEUS assignment file in order to properly model the physical processes occurring in different regions. For example, duct expansion generally leads to an increased volume for the active fuel region. Fuel and cladding are solids, and density changes are assumed to be negligible in the homogenized mixture. Therefore, the expanded volume is filled with sodium. This is modeled by increasing the volume fraction of sodium in the homogenized fuel assembly according to the volume change determined by the structural mechanics simulation.

In all other regions, the density of the region is simply multiplied by the ratio of old volume to new volume to preserve mass. In reality, an analysis such as that done for the fuel should be performed, but this could not be completed in time for this demonstration. The active fuel zone is the most important area of the core for neutronics and will have the highest effect on reactivity.

6.2 Results for Cases B and C

The eigenvalue history of the first Nek5000/PROTEUS run of both cases B and C is shown in Fig. 6.1. The case tends to converge rapidly, as shown also in Fig. 6.2, where a measure of the error (the logarithm of the difference between the current iterate and the final iterate) is shown as a function of the Nek5000/PROTEUS iteration. The peak temperature (Fig. 6.3) converges also within 10 iterations despite a “poor” initial condition. The velocity tends to converge even faster – within 2 iterations. Each Nek5000 iteration includes 1,000 time steps with a time step ranging from 0.001 s to 0.0001 s. The power is kept constant at 30 MW.

The coupled solvers led to converged results within 5 SHARP global iterations for case B. Figure 6.4 shows some of the metrics used to evaluate the dynamic behavior. Note that the eigenvalue increases significantly with the deformation. As pointed out above, the increase in eigenvalue is nonphysical and caused by the fact that densities are not updated. The model interprets the volume increase as an increase in fissile matter. It is important to note that the peak temperature does not change significantly between iterates.

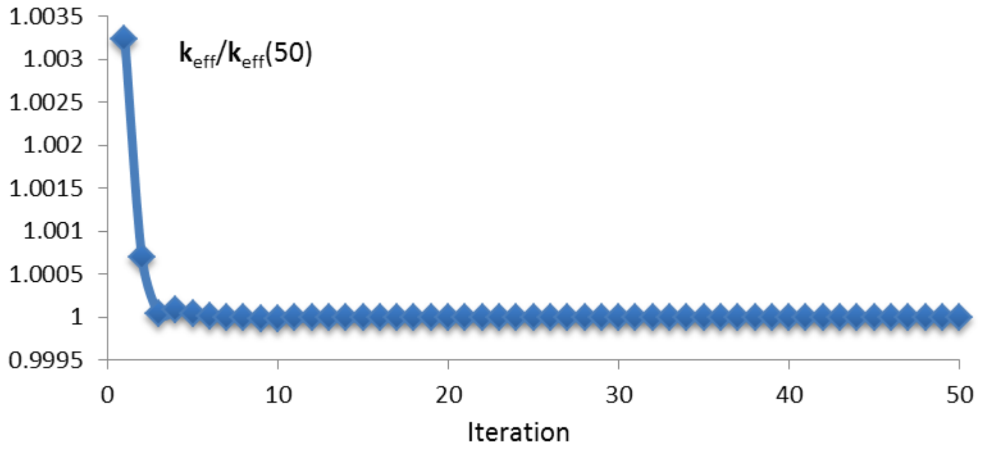


Figure 6.1 Case B eigenvalue as a function of the Nek5000/PROTEUS iteration.

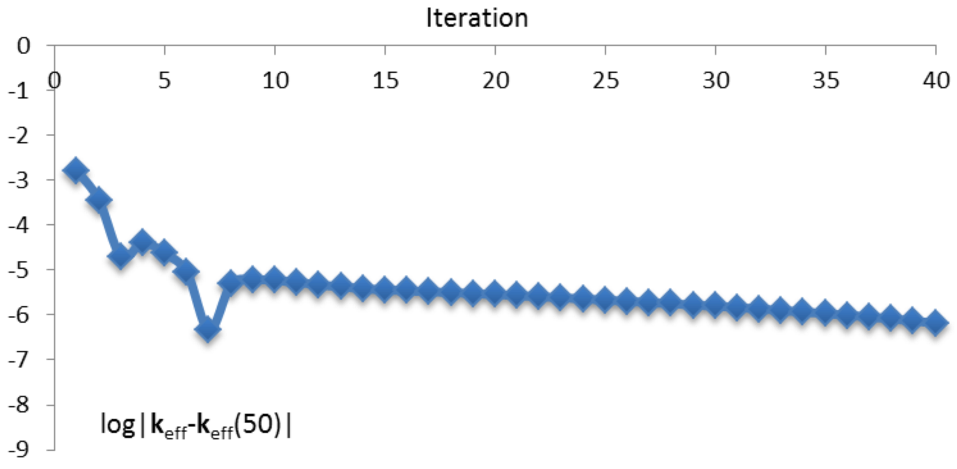


Figure 6.2 Case B error as a function of the Nek5000/PROTEUS iteration.

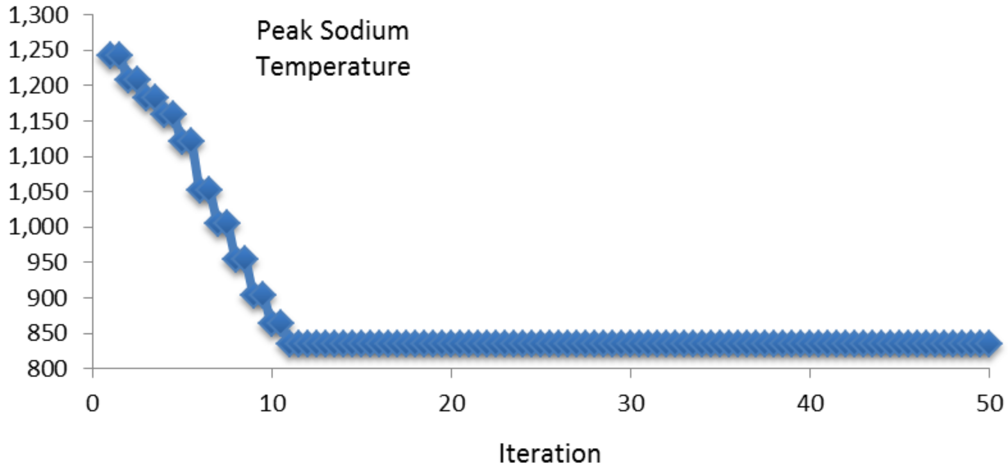


Figure 6.3 Case B Peak temperature as a function of the Nek5000/PROTEUS iteration.

This seems to indicate also that the primary mechanism of neutronic feedback is the structural mechanics, since the amplitude of eigenvalue changes due to changes in temperature (Fig. 6.3) is minimal in comparison to Fig. 6.4a.

Magnified displacements as a function of iteration are shown in Fig. 6.5. Initial and final temperature distributions are shown in Fig. 6.6. Initial and final power distributions are shown in Fig. 6.7, and the power profiles are compared in Fig. 6.8. The peak at the core centerline is slightly lower at the end of the global iteration, as expected due to overall increase in k_{eff}.

If the density changes are accounted for in PROTEUS, the behavior changes significantly (Fig. 6.9). In fact, the eigenvalue decreases significantly with the deformation, leading to feedback opposite of that shown in Fig. 6.4. Displacements and temperatures are not significantly different than those shown in Figs. 6.5 and 6.6. In case C, however, the opposite trend is observed for the power distribution as the peak at the core centerline is actually increased at the end of the global iteration.

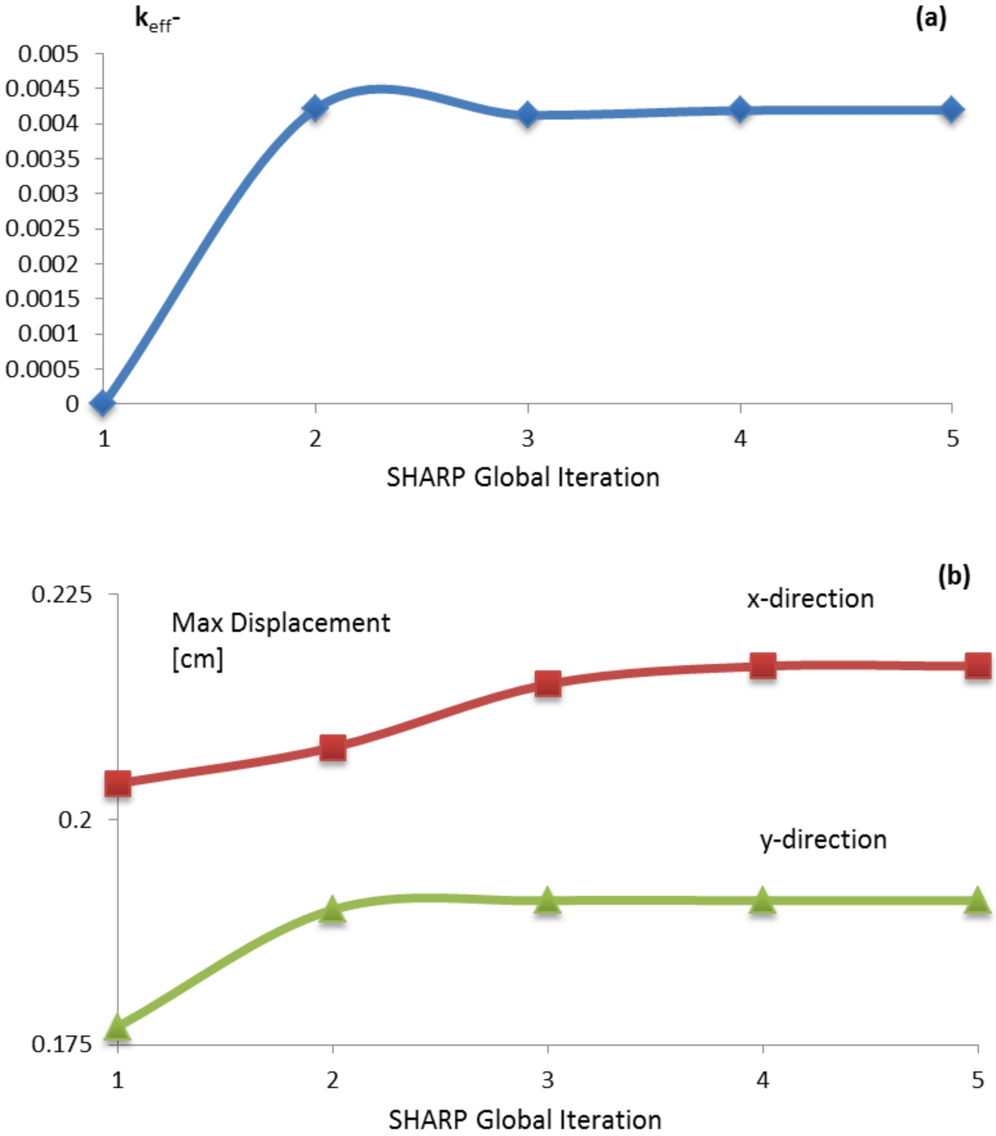
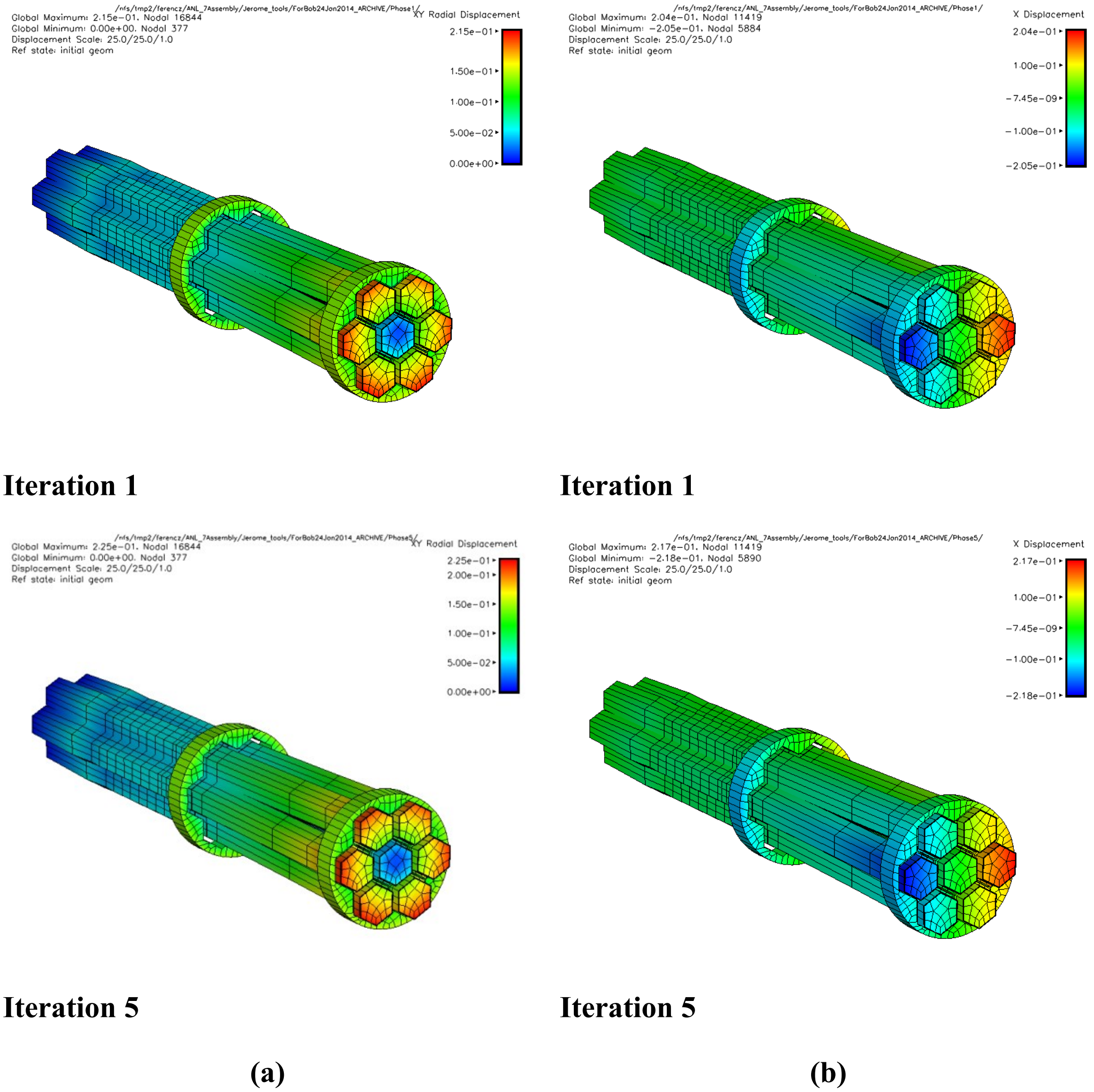


Figure 6.4 Case B results as a function of SHARP global iterations: (a) Eigenvalue difference; (b) Maximum displacement.

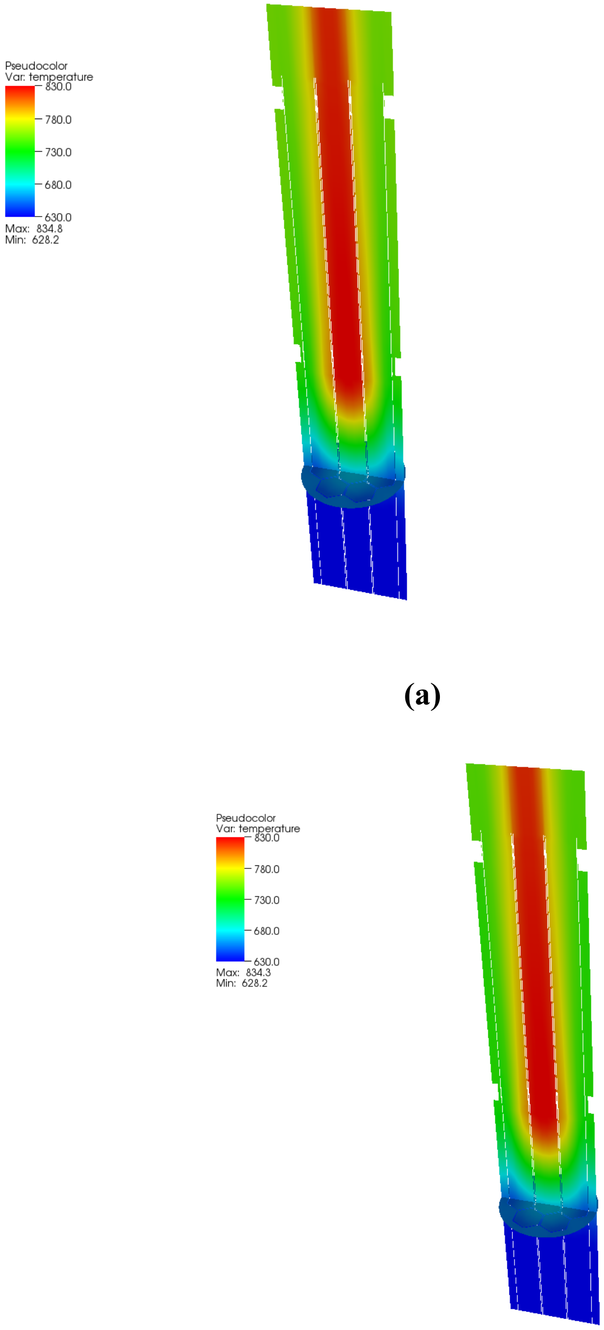

/nfs/tmp2/ferencz/ANL_7Assembly/Jerome_tools/ForBob24Jan2014_ARCHIVE/Phase1/
Global Maximum: 2.15e−01, Nodal 16844
Global Minimum: 0.00e+00, Nodal 377
Displacement Scale: 25.0/25.0/1.0
Ref state: initial geom
XY Radial Displacement
2.15e−01
1.50e−01
1.00e−01
5.00e−02
0.00e+00
/nfs/tmp2/ferencz/ANL_7Assembly/Jerome_tools/ForBob24Jan2014_ARCHIVE/Phase1/
Global Maximum: 2.04e−01, Nodal 11419
Global Minimum: −2.05e−01, Nodal 5884
Displacement Scale: 25.0/25.0/1.0
Ref state: initial geom
X Displacement
2.04e−01
1.00e−01
−7.45e−09
−1.00e−01
−2.05e−01
/nfs/tmp2/ferencz/ANL_7Assembly/Jerome_tools/ForBob24Jan2014_ARCHIVE/Phase5/
Global Maximum: 2.25e−01, Nodal 16844
Global Minimum: 0.00e+00, Nodal 377
Displacement Scale: 25.0/25.0/1.0
Ref state: initial geom
XY Radial Displacement
2.25e−01
2.00e−01
1.50e−01
1.00e−01
5.00e−02
0.00e+00
/nfs/tmp2/ferencz/ANL_7Assembly/Jerome_tools/ForBob24Jan2014_ARCHIVE/Phase5/
Global Maximum: 2.17e−01, Nodal 11419
Global Minimum: −2.18e−01, Nodal 5890
Displacement Scale: 25.0/25.0/1.0
Ref state: initial geom
X Displacement
2.17e−01
1.00e−01
−7.45e−09
−1.00e−01
−2.18e−01

Iteration 1 **Iteration 1**

Iteration 5 **Iteration 5**

(a) **(b)**

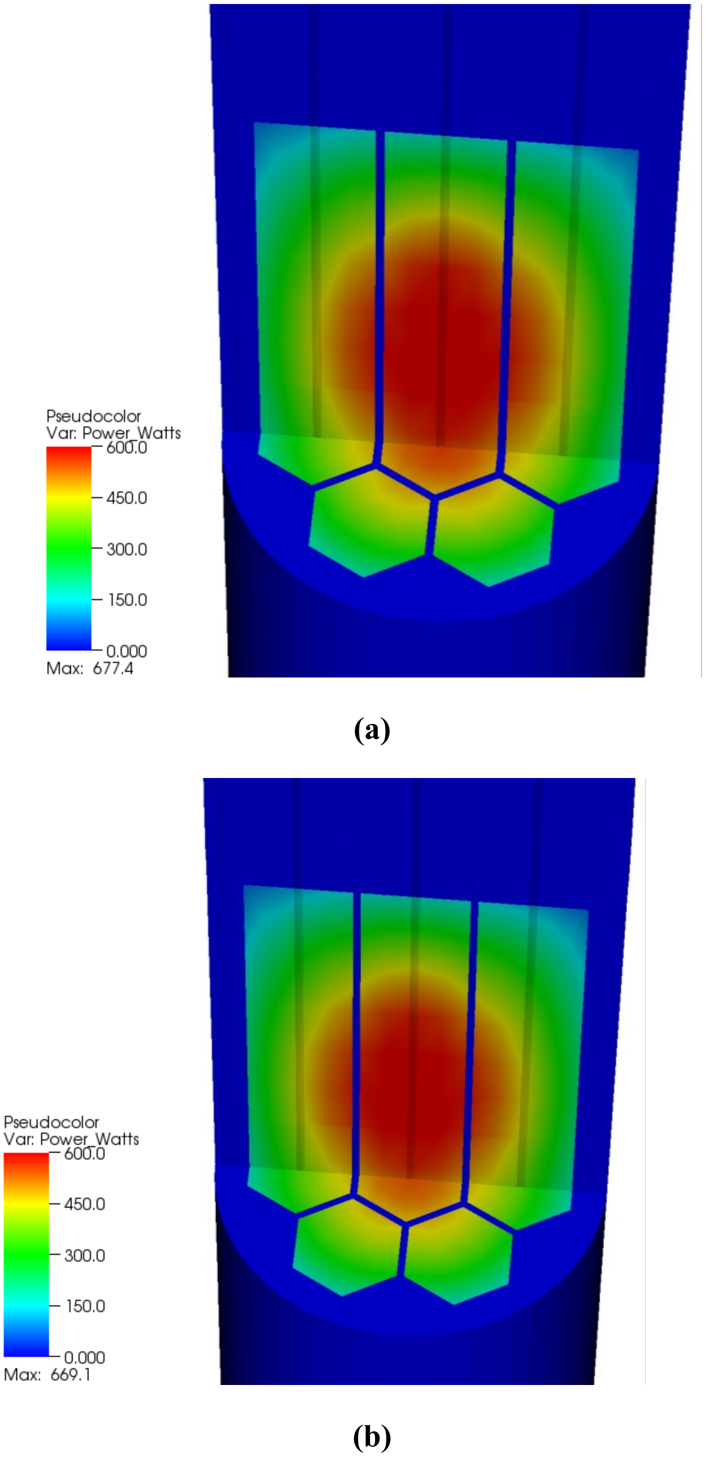
Figure 6.5 Case B displacement as a function of SHARP global iteration: (a) Displacement in the X-direction; (b) Displacement in the radial direction



(a)

(b)

Figure 6.6 Case B power distribution as a function of SHARP global iteration:(a) Iteration 1; (b) Iteration 5.



(a)

(b)

Figure 6.7 Case B Power distribution as a function of SHARP global iteration: (a) Iteration 1; (b) Iteration 5.

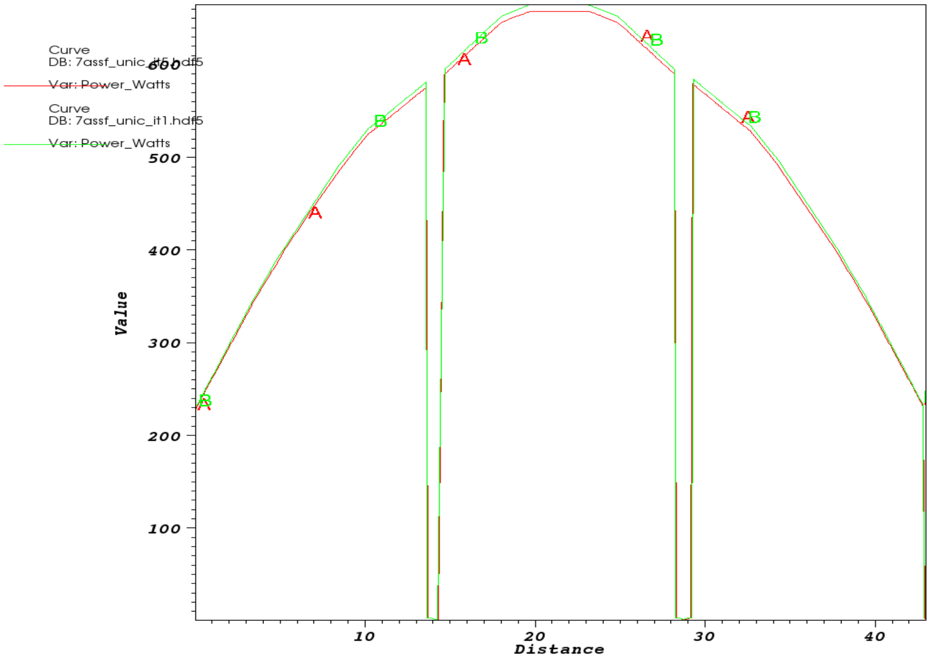


Figure 6.8 Case B power profile comparison at the midplane (z=140 cm, y=12.5 cm) (power distribution is shown in Figure 6.7).

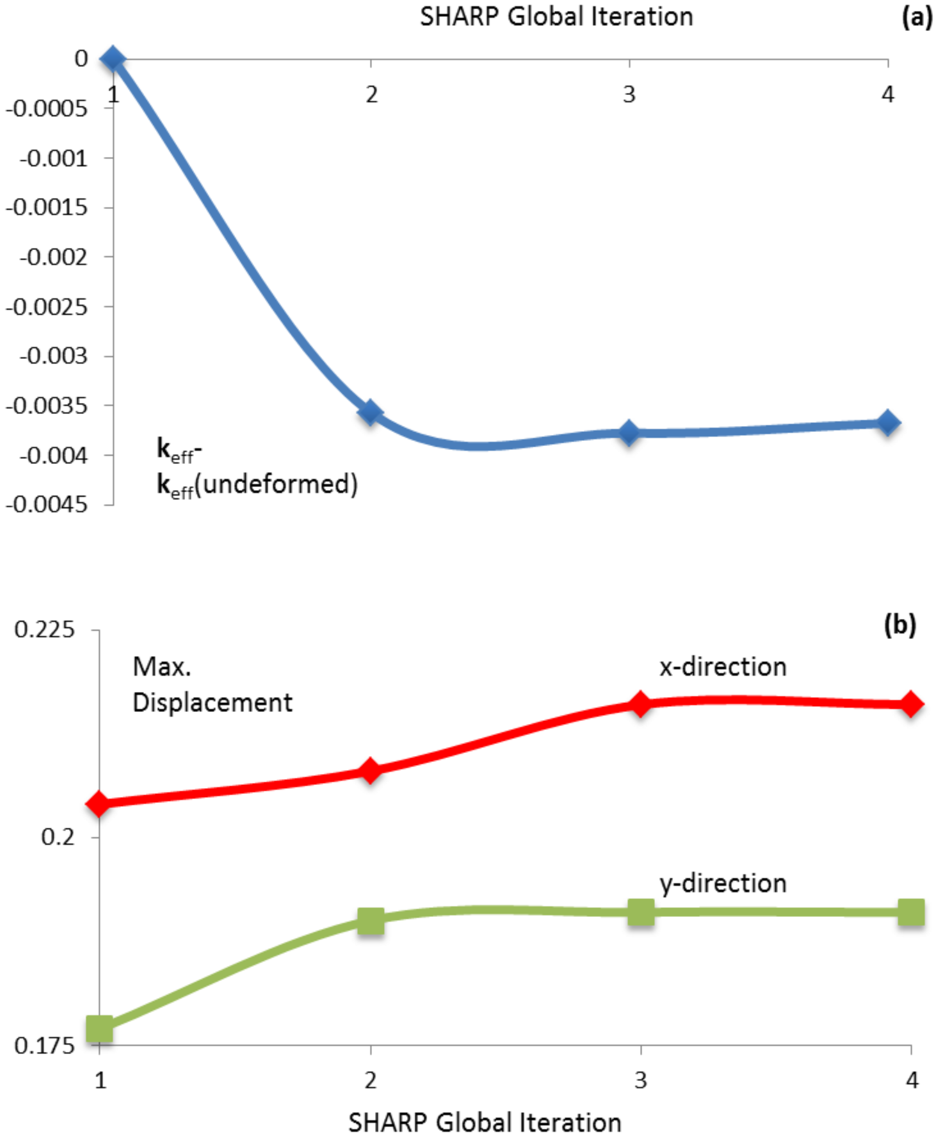


Figure 6.9 Case C eigenvalue and displacement as function of SHARP global iteration: (a) Eigenvalue difference; (b) Maximum displacement.

7 Conclusions and Future Work

The advanced nuclear reactor modeling and simulation toolkit SHARP was employed to perform a first-of-a-kind analysis of the core radial expansion phenomenon in an SFR. Physics models of a full-core model of the ABTR have also been developed for each of the three physics modules, and fully integrated quasi-static simulations of a 7-assembly mini-core test problem have been performed. Standalone results for each of the three physics modules for the ABTR are presented here, which provides a demonstration of the feasibility of the multi-physics simulation.

Because this is a first-of-a-kind simulation, a new procedure for driving the coupled simulations had to be implemented into the MOAB framework, via MBCoupler and CouPE. The future objective is to completely automate a dynamic coupling procedure, where all of the physics codes are run simultaneously under the CouPE framework. However, for this initial effort, the structural mechanics code Diablo was not yet fully integrated into CouPE, which necessitated off-line mesh mapping and deformation. This process necessitated the development of a utility called DEFORM to modify the thermal-hydraulics and neutronics meshes by applying displacements computed by Diablo.

This demonstration used the RGG tools in MeshKit for creating the meshed core models for each of the three physics codes. Certain distinctions in the meshing needs of the three physics modules motivated the development of additional features in RGG to complete this analysis. This includes the superblock feature, which enables mesh generation for the neutronics model to further subdivide material composition regions for neutron cross section evaluation, without burdening the thermal-hydraulics and structural mechanics models with the additional subdivisions. Furthermore, a feature was added to enable the user to specify the need to generate a mesh file in Exodus format, which is required by Diablo.

Models of the 7-assembly mini-core and full ABTR core were developed in all three physics modules. The MC^2-3 code was used to generate 9 energy group microscopic neutron cross sections for the PROTEUS neutronics model. The spectrum collapse procedure was performed individually for each of the homogenized compositions throughout the assemblies, as well as the explicit compositions needed for the cladding, duct walls, load pads, restraint ring, and sodium coolant. The angular domain was discretized into 96 physical angles, following the results of an angular convergence study. A new porous media model was implemented in Nek5000 in order to represent the influence of the fuel pins on the flow and temperature fields without the need to explicitly model each fuel pin. This greatly reduces the computational expense compared to performing large eddy simulations on a fine grid, and provides additional flexibility in modeling with Nek5000. Diablo is employed for the analysis of solid mechanics on the structural components, which provides a smooth displacement field to the PROTEUS and Nek5000 models. The duct walls, restraint rings, and load pads are all modeled as linearly elastic stainless steel. The option for finite elastic/plastic deformation modeling is being utilized, but in the purely elastic, temperature-independent mode. The homogenized duct interior, external sodium, and upper sodium are modeled as very soft linear elastic material.

The quasi-static coupled simulations for the 7-assembly mini-core demonstrate the modeling capability in SHARP to treat the phenomenon of radial core expansion and bowing using realistic geometry on the continuum scale. The simulations for the full ABTR core model demonstrate the feasibility of performing full-core simulations on high-performance computers available today to the DOE laboratory complex.

Future work in this area would see the completion of the ABTR core analysis, including full-scale quasi-static analysis of the core expansion and bowing. Development efforts in the CouPE framework and Diablo will enable Diablo to be fully integrated, with no need for off-line mesh mapping and deformation. This will greatly facilitate the code coupling process for analysts, and the highly efficient coupling algorithms in CouPE will accelerate simulation development and execution.

8 References

1. A. Siegel, T. Tautges, A. Caceres, D. Kaushik, P. Fischer, G. Palmiotti, M.A. Smith, J. Ragusa, “Software Design of SHARP,” in *Proceedings of the Joint International Topical Meeting on Mathematics and Computations and Supercomputing in Nuclear Applications (M&C + SNA)*, American Nuclear Society, April 2007.
2. T.J. Tautges, R. Meyers, K. Merkley, C. Stimpson, C. Ernst, *MOAB: A Mesh-Oriented Database*, Sandia National Laboratories report SAND2004-1592, April 2004.
3. M.A. Smith, D. Kaushik, A. Wollaber, W.S. Yang, B. Smith, C. Rabiti, G. Palmiotti, “Recent Research Progress on UNIC at Argonne National Laboratory,” in *Proceedings of the International Conference on Mathematics, Computational Methods and Reactor Physics (M&C),* American Nuclear Society, April 2009.
4. P.F. Fischer, J.W. Lottes, S.G. Kerkemier, Nek5000 Web Page, http://nek5000.mcs.anl.gov, 2008.
5. D. Parsons, J.M. Solberg, R.M. Ferencz, M.A. Havstad, N.E. Hodge, and A.P. Wemhoff, *Diablo User Manual*, Lawrence Livermore National Laboratory report UCRL-SM-234927, Sept. 2007.
6. Y.I. Chang, P.J. Finck, C. Grandy, *Advanced Burner Test Reactor Preconceptual Design Report*” Argonne National Laboratory report ANL-ABR-1 (ANL-AFCI-173), Sept. 2006.
7. E.R. Shemon, J. Grudzinski, C.H. Lee, J. Thomas, Specification of the Advanced Burner Test Reactor Multi-Physics Coupling Demonstration Problem, ANL/NE-15/43, Dec. 2015.
8. T.J. Tautges, H.-J. Kim, A. Caceres, R. Jain, “Coupled Multi-Physics simulation frameworks for reactor simulation: A Bottom-Up approach,” in *Proceedings of the International Conference on Mathematics and Computational Methods Applied to Nuclear Science and Engineering (M&C)*, American Nuclear Society, Rio de Janeiro, Brazil, May 2011.
9. D. Gaston, C. Newman, G. Hansen, D. Lebrun-Grandi, “MOOSE: a parallel computational framework for coupled systems of nonlinear equations,” *Nuclear Engineering and Design*, **239**(10):1768–1778, Oct. 2009.
10. D.E. Keyes et al., ”Multiphysics Simulations: Challenges and Opportunities,” *International Journal of High Performance Computing Applications*, **27**(1):4-83, 2012.
11. T. Tautges, P. Fischer, I. Grindeanu, R. Jain, V. Mahadevan, A. Obabko, M. Smith, E. Merzari, R. Ferencz, “SHARP assembly-scale multiphysics demonstration simulations,” ANL/NE-13/9, Argonne National Laboratory, Mar.2013.
12. T.J. Tautges, A. Caceres, “Scalable parallel solution coupling for multiphysics reactor simulation”, *Journal of Physics*, Conference Series, **180**, 2009.
13. G.I. Marchuk, *On the theory of the splitting-up method: Volume II of Numerical Solution of Partial Differential Equations*, Academic Press, New York, 1971.
14. D.A. Knoll, D.E. Keyes, “Jacobian-free Newton-Krylov methods: a survey of approaches and applications,” *Journal of Computational Physics*, **193**(2):357–397, 2004.
15. S. Balay, W.D. Gropp, L. Curfman McInnes, B.F. Smith, “Efficient management of parallelism in object oriented numerical software libraries,” in *Modern Software Tools in Scientific Computing*, pp. 163–202, Birkhäuser Press, 1997.

16. M.A. Smith, et al, “UNIC: development of a new reactor physics analysis tool,” in *Proceedings of Winter Meeting on International Conference on Making the Renaissance Real*, **97**:565–566, American Nuclear Society, Nov. 2007.

17. Y. Maday, A.T. Patera, “Spectral element methods for the Navier-Stokes equations,” in A.K. Noor and J.T. Oden, editors, *State-of-the-Art Surveys in Computational Mechanics*, pp. 71–143, ASME, New York, 1989.

18. A.G. Tomboulides, J.C.Y. Lee, and S.A. Orszag, “Numerical simulation of low Mach number reactive flows,” *Journal of Scientific Computing*, **12**:139–167, June 1997.

19. A.G. Tomboulides, M. Israeli, G.E. Karniadakis, “Efficient removal of boundary-divergence errors in time-splitting methods,” *Journal of Scientific Computing*, **4**:291–308, 1989.

20. *STAR-CD v4.12 Methodology Guide*, CD-adapco, Ltd., 2011.

21. R. Jain, T.J. Tautges, “NEAMS MeshKit,” presented at International Congress on the Advances in Nuclear Power Plants, Chicago, 2014.

22. T.J. Tautges, R. Jain, “Creating geometry and mesh models for nuclear reactor core geometries using a lattice hierarchy-based approach,” *Engineering with Computers*, **28**(4):319-329, 2011.

23. R. Jain, T.J. Tautges, “RGG: Reactor Geometry (and Mesh) Generator,” presented at International Congress on the Advances in Nuclear Power Plants, Chicago, 2012.

24. R. Jain, T.J. Tautges, *MeshKit*, Argonne National Laboratory report ANL/MCS-TM/336, 2013.

25. R. Jain, "Report on FY11 Extensions to MeshKit and RGG", ANL/MCS-TM-316, Argonne National Laboratory, Sep. 2011.

26. T.J. Tautges, R. Jain, “Mesh Copy/Move/Merge Tool for Reactor Simulation Applications” DOE Reactor Campaign, April 30, 2010.

APPENDIX A: INPUT FILE FOR A FUEL ASSEMBLY WITH 24 AXIAL REGIONS

```
!##############################################################################
########
! ABTR Minimal Fuel (cm).
!##############################################################################
########
Geometry Volume
GeometryType Hexagonal
!
Materials 36 MTLP MTLP &
MACLP MACLP &
Oplenum     Oplenum &
StHT9 StHT9 &
Ustructure  Ustructure &
GPNA_BC     GPNA_BC &
GPNA01      GPNA01 &
GPNA02      GPNA02 &
GPNA03      GPNA03 &
GPNA04      GPNA04 &
GPNA05      GPNA05 &
GPNA06      GPNA06 &
GPNA07      GPNA07 &
GPNA08      GPNA08 &
GPNA09      GPNA09 &
GPNA10      GPNA10 &
GPNA11      GPNA11 &
GPNA_BACLP  GPNA_BACLP &
GPNA_ACLP   GPNA_ACLP &
GPNA_BTLP   GPNA_BTLP &
GPNA_TLP    GPNA_TLP &
GPNA_ATLP   GPNA_ATLP &
NAHT9 NAHT9 &
Active01    Active01 &
Active02    Active02 &
Active03    Active03 &
Active04    Active04 &
Active05    Active05 &
Active06    Active06 &
Active07    Active07 &
Active08    Active08 &
Active09    Active09 &
Active10    Active10 &
GP_FuelBond GP_FuelBond &
GP_Bond     GP_Bond &
GP_GasBond  GP_GasBond
Duct  4     0     0     38    80.1  13.598      14.198      14.463      14.598
      NAHT9 StHT9 GPNA_BC     GPNA_BC
Duct  4     0     0     80.1  98    13.598      14.198      14.463      14.598
      NAHT9 StHT9 GPNA_BC     GPNA_BC
```

```
Duct  4     0     0     98    106   13.598      14.198      14.463      14.598
      Active01    StHT9 GPNA01      GPNA01
Duct  4     0     0     106   114   13.598      14.198      14.463      14.598
      Active02    StHT9 GPNA02      GPNA02
Duct  4     0     0     114   122   13.598      14.198      14.463      14.598
      Active03    StHT9 GPNA03      GPNA03
Duct  4     0     0     122   130   13.598      14.198      14.463      14.598
      Active04    StHT9 GPNA04      GPNA04
Duct  4     0     0     130   138   13.598      14.198      14.463      14.598
      Active05    StHT9 GPNA05      GPNA05
Duct  4     0     0     138   146   13.598      14.198      14.463      14.598
      Active06    StHT9 GPNA06      GPNA06
Duct  4     0     0     146   154   13.598      14.198      14.463      14.598
      Active07    StHT9 GPNA07      GPNA07
Duct  4     0     0     154   162   13.598      14.198      14.463      14.598
      Active08    StHT9 GPNA08      GPNA08
Duct  4     0     0     162   165.2 13.598      14.198      14.463      14.598
      Active09    StHT9 GPNA09      GPNA09
Duct  4     0     0     165.2 170   13.598      14.198      14.463      14.598
      Active09    StHT9 GPNA09      GPNA09
Duct  4     0     0     170   178   13.598      14.198      14.463      14.598
      Active10    StHT9 GPNA10      GPNA10
Duct  4     0     0     178   182   13.598      14.198      14.463      14.598
      GP_FuelBond StHT9 GPNA11      GPNA11
Duct  4     0     0     182   182.89      13.598      14.198      14.463
      14.598      GP_FuelBond StHT9 GPNA11      GPNA11
Duct  4     0     0     182.89      183.12      13.598      14.198      14.463
      14.598      GP_Bond           StHT9 GPNA_BACLP        GPNA_BACLP
Duct  4     0     0     183.12      193.28      13.598      14.198      14.463
      14.598      GP_Bond           StHT9 MACLP GPNA_ACLP
Duct  4     0     0     193.28      202.654     13.598      14.198      14.463
      14.598      GP_Bond     StHT9 GPNA_BTLP   GPNA_BTLP
Duct  4     0     0     202.654     267   13.598      14.198      14.463
      14.598      GP_GasBond  StHT9 GPNA_BTLP   GPNA_BTLP
Duct  4     0     0     267   298   13.598      14.198      14.463      14.598
      GP_GasBond  StHT9 GPNA_BTLP   GPNA_BTLP
Duct  4     0     0     298   312.72      13.598      14.198      14.463
      14.598      Ustructure  StHT9 GPNA_BTLP   GPNA_BTLP
Duct  4     0     0     312.72      322.88      13.598      14.198      14.463
      14.598      Ustructure  StHT9 MTLP  GPNA_TLP
Duct  4     0     0     322.88      328   13.598      14.198      14.463
      14.598      Ustructure  StHT9 GPNA_ATLP   GPNA_ATLP
Duct  4     0     0     328   358   13.598      14.198      14.463      14.598
      Oplenum     Oplenum     Oplenum     Oplenum
Assembly 1
XX
Center
Rotate Z 30
RadialMeshSize 0.1
AxialMeshSize 1 &
1 &
1 &
1 &
1 &
1 &
1 &
```

```
1 &
1 &
1 &
1 &
1 &
1 &
1 &
1 &
1 &
1 &
1 &
1 &
1 &
1 &
1 &
1 &
1
EdgeInterval 3
CreateSideset No
CreateMatFiles 6
List_MaterialSet_StartId 6 &
1622000      &
1818000      &
1821000      &
1916000      &
2118000      &
2219000
List_NeumannSet_StartId 6 &
1622000      &
1818000      &
1821000      &
1916000      &
2118000      &
2219000
END
```

APPENDIX B: INPUT FILE FOR A CONTROL ASSEMBLY

```
!######################################################################################
! ABTR Control Assembly (cm).
!######################################################################################
!
Geometry Volume
GeometryType Hexagonal
Materials 55 MTLP   MTLP &
MACLP  MACLP &
Oplenum      Oplenum &
StHT9  StHT9 &
Ustructure   Ustructure &
GPNA_BC      GPNA_BC &
GPNA01 GPNA01 &
GPNA02 GPNA02 &
GPNA03 GPNA03 &
GPNA04 GPNA04 &
GPNA05 GPNA05 &
GPNA06 GPNA06 &
GPNA07 GPNA07 &
GPNA08 GPNA08 &
GPNA09 GPNA09 &
GPNA10 GPNA10 &
GPNA11 GPNA11 &
GPNA_BACLP   GPNA_BACLP &
GPNA_ACLP    GPNA_ACLP &
GPNA_BTLP    GPNA_BTLP &
GPNA_TLP     GPNA_TLP &
GPNA_ATLP    GPNA_ATLP &
MNA_LR MNA_LR &
MNA_BC MNA_BC &
MNA_01 MNA_01 &
MNA_02 MNA_02 &
MNA_03 MNA_03 &
MNA_04 MNA_04 &
MNA_05 MNA_05 &
MNA_06 MNA_06 &
MNA_07 MNA_07 &
MNA_08 MNA_08 &
MNA_09 MNA_09 &
MNA_10 MNA_10 &
MNA_11 MNA_11 &
MNA_12 MNA_12 &
MNA_CP MNA_CP &
MNA_US MNA_US &
NAHT9  NAHT9 &
EmptyDuctBC  EmptyDuctBC &
EmptyDuct01  EmptyDuct01 &
EmptyDuct02  EmptyDuct02 &
EmptyDuct03  EmptyDuct03 &
EmptyDuct04  EmptyDuct04 &
EmptyDuct05  EmptyDuct05 &
EmptyDuct06  EmptyDuct06 &
EmptyDuct07  EmptyDuct07 &
EmptyDuct08  EmptyDuct08 &
EmptyDuct09  EmptyDuct09 &
Follower09   Follower09 &
Follower10   Follower10 &
Follower11   Follower11 &
```

```
Absorber11    Absorber11 &
Absorber12    Absorber12 &
CGP    CGP
Duct   6      0      0      38     80.1   12.198 12.798 13.598 14.198 14.463 14.598 NAHT9
       StHT9  MNA_LR StHT9    GPNA_BC     GPNA_BC
Duct   6      0      0      80.1   98     12.198 12.798 13.598 14.198 14.463 14.598
       EmptyDuctBC   StHT9  MNA_BC StHT9    GPNA_BC     GPNA_BC
Duct   6      0      0      98     106    12.198 12.798 13.598 14.198 14.463 14.598
       EmptyDuct01   StHT9  MNA_01 StHT9    GPNA01      GPNA01
Duct   6      0      0      106    114    12.198 12.798 13.598 14.198 14.463 14.598
       EmptyDuct02   StHT9  MNA_02 StHT9    GPNA02      GPNA02
Duct   6      0      0      114    122    12.198 12.798 13.598 14.198 14.463 14.598
       EmptyDuct03   StHT9  MNA_03 StHT9    GPNA03      GPNA03
Duct   6      0      0      122    130    12.198 12.798 13.598 14.198 14.463 14.598
       EmptyDuct04   StHT9  MNA_04 StHT9    GPNA04      GPNA04
Duct   6      0      0      130    138    12.198 12.798 13.598 14.198 14.463 14.598
       EmptyDuct05   StHT9  MNA_05 StHT9    GPNA05      GPNA05
Duct   6      0      0      138    146    12.198 12.798 13.598 14.198 14.463 14.598
       EmptyDuct06   StHT9  MNA_06 StHT9    GPNA06      GPNA06
Duct   6      0      0      146    154    12.198 12.798 13.598 14.198 14.463 14.598
       EmptyDuct07   StHT9  MNA_07 StHT9    GPNA07      GPNA07
Duct   6      0      0      154    162    12.198 12.798 13.598 14.198 14.463 14.598
       EmptyDuct08   StHT9  MNA_08 StHT9    GPNA08      GPNA08
Duct   6      0      0      162    165.2  12.198 12.798 13.598 14.198 14.463 14.598
       EmptyDuct09   StHT9  MNA_09 StHT9    GPNA09      GPNA09
Duct   6      0      0      165.2  170    12.198 12.798 13.598 14.198 14.463 14.598
       Follower09    StHT9  MNA_09 StHT9    GPNA09      GPNA09
Duct   6      0      0      170    178    12.198 12.798 13.598 14.198 14.463 14.598
       Follower10    StHT9  MNA_10 StHT9    GPNA10      GPNA10
Duct   6      0      0      178    182    12.198 12.798 13.598 14.198 14.463 14.598
       Follower11    StHT9  MNA_11 StHT9    GPNA11      GPNA11
Duct   6      0      0      182    182.89 12.198 12.798 13.598 14.198 14.463 14.598
       Absorber11    StHT9  MNA_11 StHT9    GPNA11      GPNA11
Duct   6      0      0      182.89 183.12 12.198 12.798 13.598 14.198 14.463 14.598
       Absorber12    StHT9  MNA_12 StHT9  GPNA_BACLP    GPNA_BACLP
Duct   6      0      0      183.12 193.28 12.198 12.798 13.598 14.198 14.463 14.598
       Absorber12    StHT9  MNA_12 StHT9  MACLP  GPNA_ACLP
Duct   6      0      0      193.28 202.654       12.198 12.798 13.598 14.198 14.463 14.598
       Absorber12    StHT9  MNA_12 StHT9  GPNA_BTLP     GPNA_BTLP
Duct   6      0      0      202.654       267    12.198 12.798 13.598 14.198 14.463 14.598
       Absorber12    StHT9  MNA_12 StHT9  GPNA_BTLP     GPNA_BTLP
Duct   6      0      0      267    298    12.198 12.798 13.598 14.198 14.463 14.598 CGP
       StHT9  MNA_CP StHT9  GPNA_BTLP     GPNA_BTLP
Duct   6      0      0      298    312.72 12.198 12.798 13.598 14.198 14.463 14.598
       Ustructure    StHT9  MNA_US        StHT9  GPNA_BTLP     GPNA_BTLP
Duct   6      0      0      312.72 322.88 12.198 12.798 13.598 14.198 14.463 14.598
       Ustructure    StHT9  MNA_US StHT9  MTLP   GPNA_TLP
Duct   6      0      0      322.88 328    12.198 12.798 13.598 14.198 14.463 14.598
       Ustructure    StHT9  MNA_US StHT9  GPNA_ATLP     GPNA_ATLP
Duct   6      0      0      328    358    12.198 12.798 13.598 14.198 14.463 14.598 Oplenum
       Oplenum       Oplenum       Oplenum       Oplenum       Oplenum
Assembly 1
XX
Center
Rotate Z 30
RadialMeshSize 0.1
AxialMeshSize 1 &
1 &
1 &
1 &
1 &
```

```
1 &
1 &
1 &
1 &
1 &
1 &
1 &
1 &
1 &
1 &
1 &
1 &
1 &
1 &
1 &
1 &
1 &
1 &
1
EdgeInterval 3
CreateSideset No
CreateMatFiles 10
List_MaterialSet_StartId 10 &
1521000         &
1717000         &
1720000         &
1723000         &
1919000         &
2017000         &
2020000         &
2115000         &
2121000         &
2317000
List_NeumannSet_StartId 10 &
1521000         &
1717000         &
1720000         &
1723000         &
1919000         &
2017000         &
2020000         &
2115000         &
2121000         &
2317000
END
```

APPENDIX C: INPUT FILE FOR A 7-ASSEMBLY MINI-CORE

```
!#####################################################################################
! Simple 7 assembly ABTR Core With 4 Different Assemblies and 3 Rings
!#####################################################################################
!
Geometry Volume                          ! 'Geometry' card defines if the meshes are
volume or surface
Symmetry 1                         ! 'Symmetry' card defines the desired symmetry in
the model
GeometryType HexFlat                     ! 'GeometryType' card can take values
Hexagonal Rectangular
Assemblies 7 14.598                ! 'Assemblies' card defines the number of assembly
mesh files and their pitch
1212000.exo 1212
1112000.exo 1112
1211000.exo 1211                         ! Meshfile name followed by alias
1312000.exo 1312
1113000.exo 1113
1311000.exo 1311
1213000.exo 1213
Lattice 2
    1113  1213          &
  1112  1212  1312        &
    1211  1311
NeumannSet Top  1
NeumannSet Bot  2
Background 7a_outer_ring.exo             ! Background mesh file
OutputFileName 7assm_nd_core.h5m
END
```

APPENDIX D: INPUT FILE FOR A FUEL-ONLY MINI-CORE WITH RESTRAINT RINGS

```
!#####################################################################################
! Simple 7 assembly All Fuel ABTR Mini-Core With 3 Rings
!#####################################################################################
!
Geometry Volume                          ! 'Geometry' card defines if the meshes are
volume or surface
Symmetry 1                         ! 'Symmetry' card defines the desired symmetry in
the model
GeometryType HexFlat                     ! 'GeometryType' card can take values
Hexagonal Rectangular
Assemblies 7 14.598                ! 'Assemblies' card defines the number of assembly
mesh files and their pitch
1916000.exo 1916                         ! Meshfile name followed by alias
1622000.exo 1622
2118000.exo 2118
2219000.exo 2219
1818000.exo 1818
1821000.exo 1821
1620000.exo 1620
Lattice 2
    1916  1622          &
  2118  2219  1818        &
    1620  1821
NeumannSet Top  1
NeumannSet Bot  2
Background or.exo             ! Background mesh file
OutputFileName 7a_all_fuel.h5m
END
```

APPENDIX E: INPUT FILE FOR THE FULL-CORE MESH

```
!#####################################################################################
! Full Core ABTR CORE
!#####################################################################################
!
Geometry Volume                          ! 'Geometry' card defines if the meshes are
volume or surface
Symmetry 1                         ! 'Symmetry' card defines the desired symmetry in
the model
GeometryType HexFlat                     ! 'GeometryType' card can take values
Hexagonal Rectangular
Assemblies 199 14.598                    ! 'Assemblies' card defines the number of
assembly mesh files and their pitch
1121000.exo  1121
1122000.exo  1122
..
..
199 files
Lattice 9
                 XXXX  XXXX 1327 1427 1527 1627 1727   XXXX  XXXX &
                XXXX 1226 1326 1426 1526 1626 1726 1826 1926  XXXX &
              1125 1225 1325 1425 1525 1625 1725 1825 1925 2025 2125 &
            1124 1224 1324 1424 1524 1624 1724 1824 1924 2024 2124 2224 &
          1123 1223 1323 1423 1523 1623 1723 1823 1923 2023 2123 2223 2323 &
       1122 1222 1322 1422 1522 1622 1722 1822 1922 2022 2122 2222 2322 2422 &
    1121 1221 1321 1421 1521 1621 1721 1821 1921 2021 2121 2221 2321 2421 2521 &
  XXXX 1220 1320 1420 1520 1620 1720 1820 1920 2020 2120 2220 2320 2420 2520 XXXX  &
XXXX  1219 1319 1419 1519 1619 1719 1819 1919 2019 2119 2219 2319 2419 2519 2619 XXXX &
  XXXX     1318 1418 1518 1618 1718 1818 1918 2018 2118 2218 2318 2418 2518 2618  XXXX
&
    1317 1417 1517 1617 1717 1817 1917 2017 2117 2217 2317 2417 2517 2617 2717 &
      1416 1516 1616 1716 1816 1916 2016 2116 2216 2316 2416 2516 2616 2716 &
        1515 1615 1715 1815 1915 2015 2115 2215 2315 2415 2515 2615 2715 &
          1614 1714 1814 1914 2014 2114 2214 2314 2414 2514 2614 2714  &
            1713 1813 1913 2013 2113 2213 2313 2413 2513 2613 2713 &
              XXXX 1912 2012 2112 2212 2312 2412 2512 2612 XXXX &
                XXXX  XXXX 2111 2211 2311 2411 2511 XXXX  XXXX
NeumannSet Top  1
NeumannSet Bot  2
Background outer_ring_proteus.exo
OutputFileName fc_proteus.h5m
END
```

Nuclear Engineering Division
Argonne National Laboratory
9700 South Cass Avenue, Bldg. 208
Argonne, IL 60439

www.anl.gov

Argonne National Laboratory is a U.S. Department of Energy
laboratory managed by UChicago Argonne, LLC